МИНОБРНАУКИ РОССИИ

ФГБОУ ВПО «УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра автоматизации производственных процессов

В.Г. Лисиенко
С.П. Санников

МОДЕЛИРОВАНИЕ СИСТЕМ

Методические указания
по выполнению лабораторных работ
для студентов направлений
Автоматизация технологических процессов и производств,
Управление в технических системах

Екатеринбург
2011

Печатается по рекомендации методической комиссии ЛИФ.
Протокол № 84 от 26 октября 2010 г.

Рецензент − доцент, канд. техн. наук Г.Г. Ордуянц

Редактор К.В. Корнева
Оператор компьютерной верстки Г.И. Романова

Подписано в печать 27.09.11		Поз. 16
Плоская печать	Формат 60×84 1/16	Тираж 50 экз.
Заказ №	Печ. л. 3,49	Цена 18 руб. 00 коп.

Редакционно-издательский отдел УГЛТУ
Отдел оперативной полиграфии УГЛТУ

Лабораторная работа № 1

ОСНОВНЫЕ ПРИЁМЫ РАБОТЫ В СИСТЕМЕ MATLAB

Цель работы − ознакомление со структурой и элементарными функциями системы MATLAB и приобретение основных навыков работы с её справочной и демонстрационной подсистемами.

Работа состоит из четырех частей: домашнего задания, коллоквиума, расчетно-экспериментальной части и оформления полученных результатов в виде отчета. При подготовке домашнего задания следует использовать сведения, приведённые во вводной части работы.

1.1. Краткие сведения о системе MATLAB

Система MATLAB является интерактивной системой для выполнения инженерных и научных расчётов, ориентированной на работу с массивами данных [1]. Она использует математический сопроцессор и допускает возможность обращения к программам, написанным на языках FORTRAN (родной язык первых функций этой системы) и C++. Система MATLAB имеет развитую встроенную матричную и комплексную арифметику, она ориентирована на выполнение операций с векторами и матрицами, реализует сингулярное и спектральное разложения, расчёт ранга и чисел обусловленности матриц, поддерживает работу с алгебраическими полиномами, выполняет решение нелинейных уравнений и задач оптимизации, интегрирование в квадратурах, решение дифференциальных и разностных уравнений, построение различного вида графиков, трёхмерных поверхностей и линий уровня.

В базовый набор слов системы MATLAB входят:

- спецзнаки, знаки арифметических и логических операций:

[ ] — признак многомерной переменной; ( ) — признак функции; элемент массива; **.** — разделитель в числе; признак поэлементной операции; **,** — разделитель в списке; **;** — запрет на вывод результата в рабочее окно; % — признак комментария; **!** — факториал; **:** — разделитель в ранжированной переменной; **'** — признак строки; = — операция присваивания;	+ — сложение; – — вычитание; * — умножение \ — решение системы линейных уравнений; / — деление; ^ — возведение в степень; < — меньше; > — больше; & — логическое «И»; | — логическое «ИЛИ»; ~ — логическое «НЕ»

Основной объект системы MATLAB — прямоугольный массив, который может состоять из комплексных элементов. Он может вводиться без указания размеров массива, что очень удобно в работе. Отличительная черта системы — лёгкость её модификации и адаптации к конкретным задачам пользователя. Пользователь может ввести в систему любую новую команду, оператор или функцию, которые не отличаются в дальнейшей работе от встроенных функций. Новые функции, процедуры и программы сохраняются в виде файлов с расширением .m. Это делает набор доступных операторов и функций практически неограниченным;

- специальные переменные и константы

ans - ответ в большинстве случаев; eps - точность операций с плавающей точкой; realmax - наибольшее положительное число с плавающей точкой; realmin - наименьшее положительное число с плавающей точкой; why - «поддержка» разговора с ЭВМ;	pi - 3.1415926535897… i, j - мнимая часть; inf - бесконечность; NaN - НеЧисло; isnan - истина для НеЧисла; isinf - истина для бесконечных элементов; isfinite - истина для конечных элементов; flops - счетчик операций с плавающей точкой

- тригонометрические и специальные математические функции:

Тригонометрические	**Экспоненциальные**
sin - синус; sinh - гиперболический синус; asin - арксинус; asinh - гиперболический арксинус; cos - косинус; cosh - гиперболический косинус; acos - арккосинус; acosh - гиперболический арккосинус; tan - тангенс; tanh - гиперболический тангенс; atan - арктангенс; atan2 - четырехквадрантный арктангенс; atanh - гиперболический арктангенс; sec - секанс; sech - гиперболический секанс; asec - арксеканс; asech - гиперболический арксеканс;	exp - экспонента; log - натуральный логарифм; log10 - десятичный логарифм; log2 - логарифм по основанию 2; pow2 - квадрат числа; sqrt - квадратный корень; nextpow2 - следующая степень, выше 2; **Комплексные** abs - абсолютное значение; angle - фазовый угол; conj - комплексное сопряжение; imag - мнимая часть; real - реальная часть; unwrap - фазовый угол без разрывов ($\pm\pi$); isreal - «истина» для вещественных массивов;

csc - косеканс; csch - гиперболический косеканс; acsc - арккосеканс; acsch - гиперболический арккосеканс; cot - котангенс; coth - гиперболический котангенс; acot - арккотангенс; acoth - гиперболический арккотангенс;	cplxpair - сортировка чисел в комплексные пары **Округление и работа с целыми числами** fix - округление до 0; floor - округление до +∞; ceil - округление до -∞; round - округление до ближайшего целого; mod - знаковый остаток деления; rem - остаток деления; sign - знак переменной

- элементарные матрицы и манипуляции с ними:

Элементарные матрицы	**Основные параметры массивов**
zeros - матрица 0; ones - матрица 1; eye - идентичная матрица; repmat - объединение матриц; rand - матрица случайных чисел из (0,1); randn - матрица нормально распределенных случайных чисел; linspace - вектор линейного пространства; logspace - вектор логарифмического пространства; meshgrid - двумерный массив для печати 3-D графика; **:** - выделенный вектор и индекс в матрице;	size - размер матрицы; length - длина вектора; ndims - число размерности; disp - отображение матрицы или текста; isempty - «истина» для пустых матриц; isequal - «истина» для одинаковых матриц; isnumeric - «истина» для численных массивов; islogical – «истина» для логических массивов; logical - преобразование численных значений в логические; **Манипуляции с матрицами** reshape - изменение размера; diag - диагональная матрица и выделение главной диагонали; tril - нижняя треугольная часть; triu - верхняя треугольная часть; fliplr - перевернуть слева направо; flipud - перевернуть сверху вниз; flipdim - перевернуть относительно строки; rot90 - повернуть на 90º; find - поиск индексов ненулевых элементов;
Специальные матрицы compan - матрица Фробениуса; gallery - список стандартных матриц; hadamard - Адамара; hankel - Хэнкеля; hilb - Гильберта; invhilb - обратная Гильберта; magic - магический квадрат; pascal - Паскаля; rosser - классическая проблема симметричных собственных значений; toeplitz - Теплица;	

vander - Вандермонда; wilkinson - Уилкинсона;	end - последний индекс; sub2ind - линейный индекс множества; ind2sub - множественные описания из линейного индекса

- векторные и матричные функции (линейная алгебра):

Матричный анализ	**Операции линейной алгебры**
norm - норма матрицы или вектора; normest - квадратичная норма матрицы; rank - ранг матрицы; det - определитель; trace - сумма диагональных элементов (след матрицы); null - нулевое пространство; orth - ортогонализация; rref - исключение строки в заданной форме; subspace - угол между 2-мя подпространствами;	\ and / - решение линейных уравнений; inv - обращение матрицы; cond - число обусловленности с учетом обращения; condest - число обусловленности для 1 - нормы; chol - факторизация Холесского; cholinc - неполная факторизация Холесского; lu - LU-разложение; luinc - неполное LU-разложение; qr - QR-разложение; nnls - неотрицательный метод наименьших квадратов; pinv - псевдообращение; lscov - метод наименьших квадратов с заданной ошибкой;
Матричные функции	**Собственные значения и сингулярные числа**
expm - матричная экспонента logm - матричный натуральный логарифм; sqrtm - матричный квадратный корень; funm - ожидаемая главная матричная функция;	eig - вычисление собственных чисел и собственных векторов; svd - сингулярное разложение; gsvd - обобщенное разложение; eigs - несколько собственных значений; svds - некоторые сингулярные собственные значения; poly - характеристический полином; polyeig - проблема собственных значений полинома; condeig - число обусловленности по собственным значениям; hess - форма Гессенберга;
Вспомогательные функции факторизации	
qrdelete - удаление столбца из QR-разложения; qrinsert - вставка столбца при QR-разложении; rsf2csf - из вещественной диагональной формы в комплексную; cdf2rdf - из комплексной диагональной формы в вещественную форму; balance - диагональное масштабирование для увеличения точности; planerot - вращение по Гивенсу; cholupdate - наложение 1 ранга для разложения Холесского;	

qrupdate - наложение 1 ранга для QR-разложения; **Другие матричные функции** expm1 - экспоненциальная матрица по аппроксимации Паде; expm2 - экспоненциальная матрица по рядам Тейлора; expm3 - экспоненциальная матрица по собственным значениям и собственным векторам;	qz - QZ-разложение для обобщенной проблемы собственных значений; schur - разложение по Шуру; **Особые функции** rcond - LINPACK-определитель числа обусловленности из LINPACKa

- функции работы с полиномами, интерполяция:

Интерполяция данных interp1 - 1-D интерполяция; interp1q - быстрая 1-D интерполяция; interpft - 1-D интерполяция; interp2 - 2-D интерполяция; interp3 - 3-D интерполяция; interpn - N-D интерполяция; griddata - табулирование данных; **Сплайн-интерполяция** spline - кубическим сплайном; ppval - кусочно-полиномиальная; **Дополнительные функции** icubic - 1-D-кубическая интерполяция; interp4 - 2-D-линейная интерполяция; interp5 - 2-D-кубическая интерполяция; interp6 - 2-D интерполяция; table1 - 1-D поиск в таблице; table2 - 2-D поиск в таблице;	**Полиномиальные функции** roots - корни полинома; poly - преобразование корней в полином; polyval - вычисляет полином; polyvalm - вычисляет полином с матричными аргументами; residue - разложение на простые дроби; polyfit - подгонка полинома под данные polyder - дифференцирование полинома; conv - умножение полиномов; deconv - деление полиномов; **Геометрический анализ** delaunay - триангуляция по Делайну; dsearch - поиск ближайшей точки полигона; tsearch - поиск наибольшего закрытого треугольника в полигоне; voronoi - диаграмма Вороного inpolygon - принадлежность точки полигону rectint - область пересечения прямоугольников polyarea - площадь многоугольника

В системе MATLAB имеется необходимый набор команд для управления интерактивным процессом. Это команды:

- получение общих сведений о системе:
  - **helpwin** - диалоговая помощь, отдельное окно для просмотра;
  - **helpdesk** - полная документация в виде гипертекста;
  - **demo** - запуск демонстрационной системы;
  - **ver** - MATLAB, SIMULINK и приложения – информация о версиях;
  - **whatsnew** - отображение файлов Readme;
  - **readme** - что нового в MATLAB;
- работа с оперативной памятью системы (рабочим пространством):
  - **who** - список текущих переменных;
  - **whos** - список текущих переменных, полная форма;
  - **clear** - удаление переменных и функций из памяти;
  - **pack** - оптимизация (дефрагментация) рабочего пространства;
  - l**oad** - загрузка переменных рабочего пространства с диска;
  - **save** - сохранение переменных рабочего пространства на диске;
  - **quit** - окончание сессии MATLAB;
- функции и команды управления:
  - **what** - список определенных в MATLAB файлов в заданном каталоге;
  - **type** - список М-файлов;
  - **edit** - редактирование М-файлов;
  - **lookfor** - поиск всех М-файлов по ключу;
  - **which** - определение пути к функции или файлу;
  - **pcode** - создание предварительного псевдокода файла (Р-файл);
  - **inmem** - список функций в памяти;
  - **mex** - компиляция MEX-функций;
- команды настройки и поиска путей к файлам:
  - **path** - искать/установить путь;
  - **addpath** - добавить каталог в список путей;
  - **rmpath** - удалить каталог из списка путей;
  - **editpath** - корректировать список путей;
- команды управления рабочим окном системы:
  - **echo** - повторение команд в М-файлах;
  - **more** - управление постраничным выводом в командном окне;
  - **diary** - ведение протокола сессии MATLAB и его сохранение;
  - **format** - установка формата вывода;
- команды управления операционной системой компьютера:
  - **cd** - смена текущего рабочего каталога;
  - **copyfile** - копирование файлов;
  - **pwd** - показ (печать) текущего каталога;
  - **dir** - список файлов каталога;
  - **delete** - удаление файла;
  - **getenv** - просмотр переменных окружения;
  - **mkdir** - создание каталога;
  - **!** - выполнение команды операционной системы;

dos - выполнение команд DOS с возвратом результата;
web - открытие Web-администратора для доступа к сайтам или файлам;
computer - тип компьютера.

С системой MATLAB поставляется свыше сотни м-файлов, которые содержат демонстрационные примеры. Изучение этих примеров и открытая архитектура системы значительно облегчают первоначальное, а затем и профессиональное изучение ядра системы и её многочисленных приложений:

- главный демонстрационный файл

demo - просмотр и выбор демофайла MATLAB, Toolboxes и SIMULINK;

- составляющие демонстрационного файла:

intro - введение в основные матричные операции MATLAB;
inverter - демонстрация обращения матриц;
buckydem - граф Buckminster Fuller геодезического домена;
sparsity - демонстрация эффектов с разреженными матрицами;
matmanip - введение в матричные операции;
eigmovie - решение проблемы симметричных собственных значений;
rrefmovie - вычисление редуцированного ряда Echelon;
delsqdemo - конечные разности лапласиана переменных областей;
sepdemo - демонстрация системы конечных элементов;
airfoil - отображение разреженных матриц в виде аэрообъектов NASA;
eigshow - графическая демонстрация собственных значений матриц;
svdshow - графическая демонстрация сингулярных значений матриц;

1.2. Домашнее задание

1. Ознакомиться с набором операций и функций, приведенных в кратком описании системы MATLAB.
2. Составить последовательность обращения к функциям системы MATLAB при вводе экспериментальной нелинейной зависимости $Y=f(X)$, в процессе выполнения которой:

а) строится её график;

б) проводится её аппроксимация полиномами степеней с первого по третий;

в) вычисляются по найденным коэффициентам три аппроксимации;

г) вычисленные аппроксимации накладываются на график исходной зависимости.

3. Составить последовательность обращения к функциям системы MATLAB при работе с матрицами:

а) задать магическую матрицу четвертого порядка;

б) найти сумму элементов матрицы по строкам, столбцам и диагонали;

в) вычислить обратную матрицу и произведение прямой и обратной матриц;

г) найти собственные значения и собственные векторы прямой и обратной матриц;

д) вычислить экспоненту и логарифм прямой и обратной матриц.

4. Продумать порядок просмотра демонстрационных программ системы MATLAB.

1.3. Лабораторное задание

1. Ознакомиться с рабочим окном системы MATLAB (версия 6.x). Просмотреть состав команд **File, View, Edit, Windows, Help**.
2. Запустить процедуру **demo** и посмотреть основные матричные операции и манипуляции с матрицами во вкладке **Matrices**. Посмотреть также все процедуры вкладок **Numerics** и **Visualization**. Записать примеры последовательностей команд, необходимые для выполнения последующих пунктов лабораторного задания.
3. Настроить числовой формат по команде **File/Preferences**, вкладка **General**. Выполнить действия с константами, числами и элементарными математическими функциями. Записать значение погрешности вычислений **eps**. Правила вызова и использования функций можно найти в их описаниях, которые выводятся на экран, если набрать команду **help,** и через пробел имя функции.
4. Посмотреть графики специальных функций:

Бесселя – **besselj**, **bessely**, **besseli**, **besselk**;

гамма-функции – **gamma**, **gammainc**, **gammaln**.

Для построения графиков использовать процедуры построения графиков **plot** и **ezplot**.

5. Выполнить п. 2 домашнего задания.
6. Выполнить п. 3 домашнего задания.

1.4. Содержание отчёта

1. Цель работы.
2. Описание использованных в лабораторной работе функций системы MATLAB: назначение, входные и выходные параметры, пример применения.
3. Отредактированный протокол работы по выполнению п. 3 лабораторного, а также п. 2 и п. 3 домашнего задания. Для запоминания последовательности действий можно использовать команду **diary** или окно редактирования m-файлов (**File / Open / M-file**).
4. Выводы по работе. Достоинства и недостатки системы MATLAB (по сравнению с системой MathCAD).

1.5. Контрольные вопросы

1. Каким образом задаётся поэлементное выполнение арифметических операций в матрицах?
2. Назовите целочисленные функции системы MATLAB.
3. Приведите форму записи комплексного числа и основные функции при работе с комплексными числами.
4. Коэффициенты некоторого полинома получены для точки разложения, равной нулю (разложение Маклорена). Как получить коэффициенты этого полинома в системе MATLAB для другой точки (разложение Тейлора)?
5. Составьте m-файл для вычисления биномиальных коэффициентов.
6. Что означают функции: **max(V)**, **min(V)**, **mean(V)**, **std(V)**, **sort(V)**, **sum(V)**, **prod(V)**, **cumsum(V)**, **cumprod(V)**, **dif(V)**, где **V** – вектор данных.
7. Как изменятся результаты работы этих функций при подстановке вместо вектора **V** матрицы **A**?
8. Дайте определение рабочего пространства и рабочей области. В чём отличие рабочего пространства от оперативной памяти реального компьютера?
9. В чём отличие операции, обозначаемой символом «/», от операции, обозначаемой символом «\»?
10. Поясните методику изображения графиков в MATLAB. Составьте функцию для рисования графиков двух зависимостей разной длины.
11. В чём отличие команд **eval** и **feval**?

Лабораторная работа № 2

МОДЕЛИРОВАНИЕ ЭЛЕМЕНТОВ СИСТЕМ

Цель работы − ознакомление с численными методами аппроксимации экспериментальных характеристик элементов систем и современными средствами их реализации на примере определения параметров нелинейного безынерционного прибора.

Работа состоит из четырех частей: домашнего задания, коллоквиума, расчетно-экспериментальной части и оформления полученных результатов в виде отчета. При подготовке домашнего задания следует использовать теоретические сведения, приведённые в начале работы.

2.1. Описание алгоритма вычислительной задачи

2.1.1. Аппроксимация нелинейных характеристик безынерционных элементов

В радиотехнике [2] часто используются нелинейные безынерционные элементы с характеристикой

$$i = f(u), \qquad (2.1)$$

где u – напряжение и i – ток связаны вольт-амперной характеристикой (ВАХ) элемента. Типичным примером такого элемента является полупроводниковый диод.

Если на входе элемента действует гармоническое напряжение:

$$u(t) = U_{см} + U_m \cos \omega_0 t \,, \tag{2.2}$$

где $U_{см}$ – напряжение смещения (постоянное напряжение в рабочей точке); U_m – амплитуда гармонического сигнала частоты ω_0 на входе нелинейного элемента, то ток $i(t)$ в силу нелинейности характеристики существенно отличается от гармонического и может быть разложен в ряд по гармоникам частоты ω_0:

$$i(t) = I_{0к} + \sum_{n=1}^{\infty} I_{nк} \cos(n\omega_0 t + \varphi_n) \,. \tag{2.3}$$

Для определения параметров разложения ($I_{nк}$, φ_n) в (2.3) можно использовать степенную, кусочно-линейную и экспоненциальную аппроксимации входной характеристики.

Степенная аппроксимация применяется при исследовании входных характеристик на небольших участках (слабых сигналах):

$$i = a_0 + a_1 u + A_2 u^2 + \ldots + a_n u^n \tag{2.4}$$

Зависимость (2.4) получена для нулевого смещения. Если напряжение смещения отлично от нуля, то обычно $f(u)$ раскладывают в ряд Тейлора в рабочей точке:

$$i = a_0 + a_1 \cdot (u - U_{см}) + a_2 (u - U_{см})^2 + \ldots \tag{2.5}$$

Коэффициенты $\{a_k\}$, k=0...n можно определить методом наименьших квадратов по формуле

$$a = \Phi^{-1} \phi^T I \,, \tag{2.6}$$

где $\phi = \begin{bmatrix} 1 & u_1 & \ldots & u_1^n \\ 1 & u_2 & \ldots & u_2^n \\ \ldots & \ldots & \ldots & \ldots \\ 1 & u_r & \ldots & u_r^n \end{bmatrix}$ – матрица степеней отсчётов напряжения в r точках ВАХ (матрица Вандермонда),

$$\Phi = \phi^T \cdot \phi \,, \tag{2.7}$$

$I = \begin{bmatrix} i_1 & i_2 & \ldots & i_r \end{bmatrix}^T$ – вектор отсчётов тока в r точках ВАХ.

Количество точек r на реальной ВАХ прибора должно быть больше, чем порядок полинома плюс один (n+1), и редко превышает 15−20.

Кусочно-линейная аппроксимация наиболее часто используется при больших входных сигналах:

$$i_{кла} = \begin{cases} S(u - U_0), & u > U_0, \\ 0, & u < U_0, \end{cases} \tag{2.8}$$

где S - крутизна ВАХ; U_0 - напряжение отсечки. Определение крутизны S и напряжения отсечки U_0 – двух параметров этой аппроксимации – также может быть выполнено методом наименьших квадратов:

$$\sum_{p=1}^{r}(i_p - i_{кла}(u_p))^2 = \sum_{p=1}^{r}(i_p - S\cdot(u_p - Uo)\cdot 1(u_p - Uo))^2 \to \min, \qquad (2.9)$$

где $1(u)$ – единичная функция.

Экспоненциальная аппроксимация применяется при точных расчётах в режиме больших сигналов:

$$i_{экс} = i_0(e^{aU} - 1). \qquad (2.10)$$

2.1.2. Структурная схема приложения

На рис. 2.1 изображена схема связей между функциями приложения. С её помощью описывается многоуровневость проекта.

Меню первого уровня

GlMenu.m – модуль приложения, содержащий главное меню. Организация запуска одного из m-файлов DanMenu, TipApp. Входных и выходных параметров нет. Пример этого модуля показан ниже в п. 2.1.3. Там же приведены описания параметров приложения.

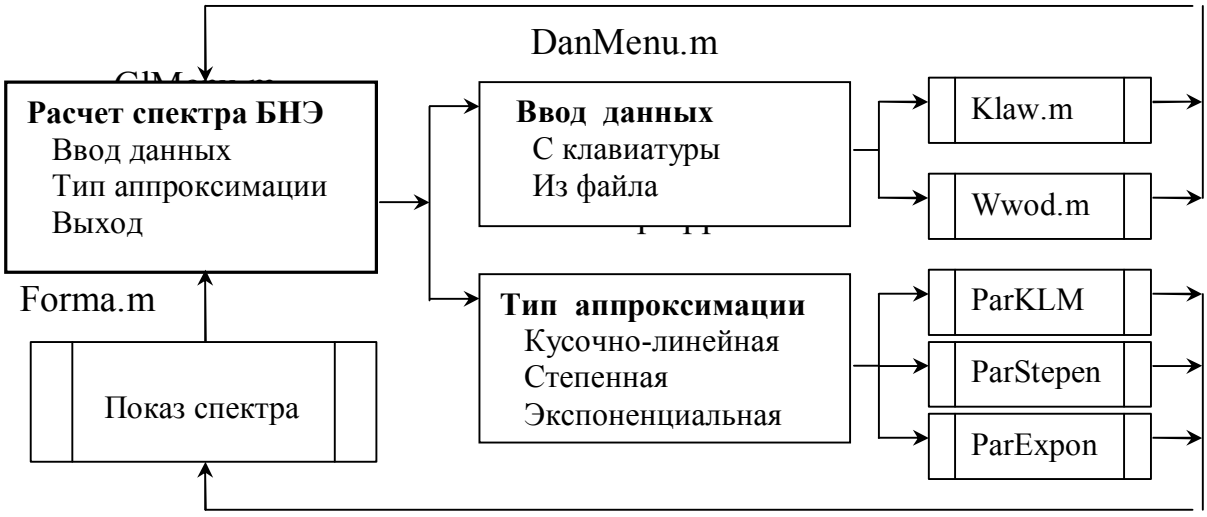

Рис. 2.1. Схема связей между функциями проекта

Меню второго уровня

TipApp.m – модуль выбора типа аппроксимации: вызов одного из m-файлов ParKLM, ParStepen, ParExpon. Входных и выходных параметров нет. Пример этого модуля показан ниже в п. 2.1.3.

Модули вычислений и построения графиков

Файлы ParKLA.m, ParStepen.m, ParExpon.m содержат алгоритмы определения параметров трёх типов аппроксимации. Forma.m – показ системы теоретических графиков с параметрами в текстовом окне. Параметры

моделей инициированы в GlMenu.m и изменены в результате вычислений в зависимости от типа аппроксимации. Входных и выходных параметров в файле Forma.m нет, однако при работе модуля используются переменные приложения, хранящиеся в рабочем пространстве. Его содержимое показано ниже. Оно может быть дополнено показом графиков кусочно-линейной (красного цвета) и степенной (синего цвета) аппроксимаций теоретической ВАХ.

2.1.3. Примеры составления программных m-файлов [1]

Файл GlMenu.m

```
% GlMenu - главное меню проекта
% Задаются глобальные параметры (масштаб - 1 вольт, 1 миллиампер)
% D1=[0.7 150];      Вектор данных кусочно-линейной [U0 S]
% D2=[0 1 2 3 4 5 6 7];          Вектор данных степенной [a0 a1 a2 a3 a4
a5 a6 a7]
% D3=[1e-12 0.045];         Вектор данных экспоненциальной [I0, a]
% Выбирается одна из задач проекта: Ввод данных, Тип аппроксимации

NumApp=0;                 %  Номер типа аппроксимации
PrUp=0;                      %  Признак полноты ввода данных
Np=2;                        %  Начальный порядок полинома
D1=[0.7 150];
D2=[0 1 2 3 4 5 6 7];
D3=[1e-12 0.045];
k=0;
while k == 0
  flag=menu('Расчёт спектра БНЭ','Ввод данных','Тип
аппроксимации','Выход');
  switch flag
    case 1
        Wwod
    case 2
      if PrUp==1
        TipApp
      end
    case 3
      return
  end
  k=0;
end

% Ввод параметров кусочно-линейной модели
      D1=curvefit('KLA',D1,up,ip);
      plot(up,ip,up,KLA(D1,up))
      return
```

Файл TipApp.m

```
% TipApp - функция выбора типа аппроксимации.
% Создаёт на экране меню выбора и направляет процесс в
% один из трёх модулей аппроксимации

m2=menu('Тип  аппроксимации', ...
  'Кусочно-линейная', ...
  'Степенная', ...
  'Экспоненциальная');
  switch m2
    case 1
       ParKLA
      case 2
        ParStepen
    case 3
       ParExpon
  end
```

Файл Wwod.m

```
% Ввод параметров программы D1,D2,D3,D4
% из файла Nabor1.dat
% и значений реальной ВАХ диода
% из файла BAX1.dat

title='Ввод данных программы из файла';
prompt={'Укажите имя файла (без расширения dat)'};
lines= 1;
def={'Nabor1'};
answer= inputdlg(prompt,title,lines,def);
s=strcat(char(answer),'.dat');
if ~isempty(s)
   fid = fopen(s,'rt');
   D1=fscanf(fid,'%e  %e',2);
   fgets(fid);
   D2=fscanf(fid,'\n%e  %e  %e  %e  %e  %e  %e  %e',8);
   fgets(fid);
   D3=fscanf(fid,'%e  %e',2);
   fgets(fid);
   D4=fscanf(fid,'%e  %e  %e',3);
   fclose(fid);
 end
 title='Ввод данных вольт-амперной характеристики ';
 prompt={'Укажите имя файла (без расширения dat)'};
```

```
 lines= 1;
 def={'BAX1'};
 answer= inputdlg(prompt,title,lines,def);
 s=strcat(char(answer),'.dat');
 if ~isempty(s)
    fid = fopen(s,'rt');
    BAX1=fscanf(fid,'%e  %e',[2 inf]);
    k=1;
    u=BAX1(1,:); i=BAX1(2,:);
    for l=1:length(u)
      if u(l)>= 0
        up(k,1)=u(l);
        ip(k,1)=i(l);
        k=k+1;
      end
    end
    fclose(fid);
    PrUp=1;
 end
return
```

Файл ParKLA.m

```
%Определение параметров кусочно-линейной модели
% и построение зависимостей
D1=curvefit('KLA',D1,up,ip);
plot(up,ip,up,KLA(D1,up))
return
```

Файл KLA.m

```
function F=KLA(y,x)
n=length(x);
for k=1:n
   if x(k) >= y(2)
     F(k,1)=y(1)*(x(k)-y(2));
   else
     F(k,1)=0;
  end
end
```

Файл Teor.m

```
% Teor.m - показ системы графиков с параметрами в текстовом окне
% Глобальные параметры инициированы в GlMenu.m и изменены в % результате расчётов в зависимости от типа аппроксимации
        k1=9;    %Количество гармоник спектра
```

```
Umin=min(up(1),D4(1)-D4(2));
Umax=max(up(length(up)),D4(1)+D4(2));
up1=Umin:(Umax-Umin)/100:Umax;
switch NumApp
  case 1, ip1=KLA(D1,up1);
  case 2, ip1=polyval(D2(8-Np:8),up1);
  case 3, ip1=D3(1)*(exp(up1/D3(2))-1);
end
subplot(2,3,1);
plot(up,ip,up1,ip1),xlim([Umin Umax]);grid;
%Вывод графика напряжения на входе
subplot(2,3,4);
t=0:0.01:10;
s1=D4(1)+D4(2)*cos(D4(3)*(pi/4)*t);
plot(s1,t),xlim([[Umin Umax]]);grid;
%Вывод графика тока БНЭ
subplot(2,3,2);
switch NumApp
  case 1, s2=KLA(D1,s1);
  case 2, s2=polyval(D2(8-Np:8),s1)';
  case 3, s2=D3(1)*(exp(s1/D3(2))-1)';
end
plot(t,s2),grid;
subplot(2,3,5);
%Вывод спектра тока на выходе БНЭ
for k=0:k1
   Sc=sum(cos(k*D4(3)*(pi/4)*t)*s2);
   Ss=sum(sin(k*D4(3)*(pi/4)*t)*s2);
   Sw(k+1)=sqrt(Sc^2+Ss^2)/length(t);
end
k=0:k1;
stem(k,Sw(1:k1+1)),grid;
%plot(k,abs(Sw1)),grid;
%Создание текстового окна
subplot(2,3,[3,6]);
axis('off');
%Вывод данных в текстовое окно
sprogram='BNE'; sname='Трухин М.П.';bnename=s;
hl=text(0.3,1.05, 'БНЭ','FontSize',14);
hl=text(0.0,0.98, 'Входной файл','FontSize',12);
hl=text(0.2,0.93, bnename,'FontName','New Times','FontSize',12);
hl=text(-0.1,0.82,'Вид аппроксимации','FontSize',12,'FontWeight','Bold');
hl=text(0.0,0.74, 'Кусочно-линейная','FontSize',12);
```

```
hl=text(-0.2,0.68, sprintf('Uo = %g B',D1(2)),'FontSize',10);
hl=text(0.68,0.68, sprintf('S = %g мА/В',D1(1)),'FontSize',10);
hl=text(0.1,0.61, 'Степенная','FontSize',12);
hl=text(-0.1,0.55, sprintf('ao = %g ',D2(1)),'FontSize',10);
hl=text(0.8,0.55, sprintf('a4 = %g ',D2(5)),'FontSize',10);
hl=text(-0.1,0.50, sprintf('a1 = %g ',D2(2)),'FontSize',10);
hl=text(0.8,0.50, sprintf('a5 = %g ',D2(6)),'FontSize',10);
hl=text(-0.1,0.45, sprintf('a2 = %g ',D2(3)),'FontSize',10);
hl=text(0.8,0.45, sprintf('a6 = %g ',D2(7)),'FontSize',10);
hl=text(-0.1,0.40, sprintf('a3 = %g ',D2(4)),'FontSize',10);
hl=text(0.8,0.40, sprintf('a7 = %g ',D2(8)),'FontSize',10);
hl=text(0.0,0.33, 'Показательная','FontSize',12);
hl=text(-0.2,0.27, sprintf('Io = %g мА',D3(1)),'FontSize',10);
hl=text(0.75,0.27, sprintf('Fi = %g B',D3(2)),'FontSize',10);
hl=text(0.0,0.18, 'Входной сигнал','FontSize',12,'FontWeight','Bold');
hl=text(0.2,0.12, sprintf('Usm = %g ',D4(1)),'FontSize',10);
hl=text(0.2,0.07, sprintf('Um = %g ',D4(2)),'FontSize',10);
hl=text(0.2,0.02, sprintf('F = %g ',D4(3)),'FontSize',10);
hl=text(0.1,-0.05, ['Программа  ' sprogram],'FontSize',10);
tm=fix(clock); Tv=tm(4:6);
hl=text(-0.15,-0.09, [sprintf(' %g :',Tv) '  ' date ],'FontSize',10);
return
```

После запуска из командной строки файла GlMenu.m на экране монитора появляется изображение, показанное на рис. 2.2.

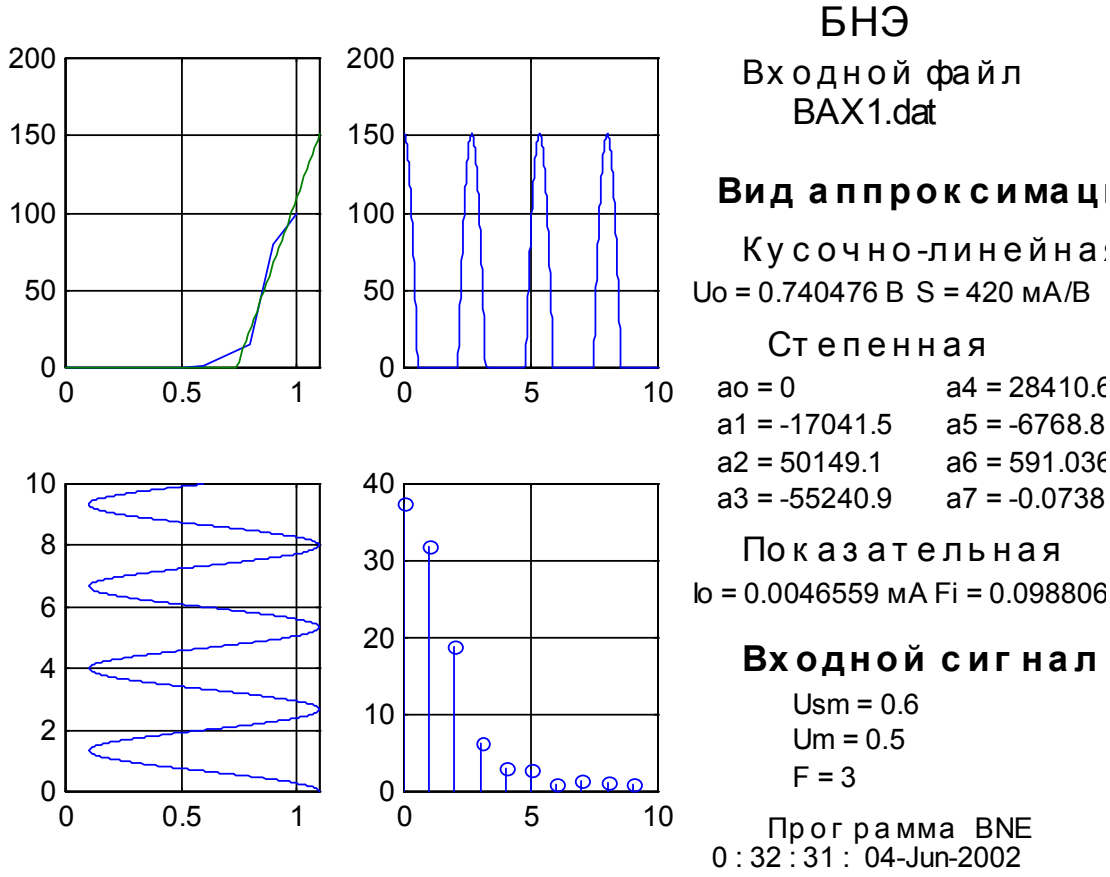

Рис. 2.2. Вид окна с изображением результатов аппроксимации

2.1.4. Программа аппроксимации характеристик CurveExpert

CurveExpert позволяет обрабатывать в одном запуске большое количество файлов данных. Данные представляются в виде столбцов, столбцы могут быть добавлены к набору данных, их можно переставлять. При одномерной зависимости файл содержит два столбца данных.

На левой стороне диалогового окна имеется список выбираемых пользователем данных (файлов). Указывается первый столбец – аргумент – и количество считываемых столбцов (функций). При отсутствии аргумента выбором команды «Автозаполнение» независимая переменная автоматически дополняется линейной последовательностью. Первый номер (число) в этой последовательности определен в поле «Минимум», приращение определено в поле «Приращения». Длина столбца указывается в поле «Количество точек». Если всегда читаются простые файлы текста с двумя столбцами, то этот диалог может быть подавлен. Чтобы восстанавливать доступ к этому диалогу, следует выбрать «Tools|Options» и выполнить необходимую настройку.

Программа CurveExpert выполняет перечисленные ниже виды аппроксимации (команда «Data|Manipulate»).

Четыре типа сплайнов: линейный, квадратный, кубический, напряженный. Линейный сплайн - самая простая форма сплайнов - просто заменяет диапазон между точками данных прямой линией. Данные должны быть отсортированы. Квадратный сплайн интерполирует диапазон между точками данных квадратичным полиномом. Сплайн проходит по точкам, и его первая производная является непрерывной. Он лучше, чем линейный сплайн, но часто далеко отклоняется от точек данных. Квадратный сплайн должен иметь одно свободное условие. Программа CurveExpert определяет, что вторая производная первого участка нулевая, что означает соединение первых и вторых точек прямой линией. Кубический сплайн интерполирует диапазон между точками данных кубичными полиномами. Кубическая аппроксимация гарантирует, что каждый сплайн проходит по точкам данных, первые и вторые производные непрерывны в узлах. Вторая производная на концах сплайна равна нулю. Напряженный сплайн ведет себя подобно кубическому с той разницей, что имеется дополнительный параметр, который определяет, как «сильно» сплайн деформируется в каждом узле. Низкий параметр напряженности (0,1) делает его подобным кубическому, высокий (20) – линейному сплайну.

Метод Lagrangian просто интерполирует набор данных полиномом. Порядок полинома задается числом точек в наборе данных - полином должен иметь достаточно степеней свободы для аппроксимации. Для набора данных с n точками полином будет иметь порядок n-1. Степенная аппроксимация особенно хороша для наборов данных с малым числом точек (8 или меньше). Точки не должны слишком рассеиваться. Подобно всем полиномам, интерполяция по Лагранжу усиливает колебательное поведение при увеличении степени. В настоящее время программа CurveExpert ограничивает использование этого типа аппроксимации 13 точками или меньше.

Линейные модели, которые состоят из линейной комбинации заданного набора функций, и линейный регресс используются, чтобы минимизировать различие между моделью и данными. Общая форма этого вида моделей:

$$Y(t) = \sum_k a_k X_k(t),$$

где a_k – искомые параметры; $X_k(t)$ – базовые функции.

Определение «линейный» относится только к зависимости модели от параметров a_k; функции $X_k(t)$ могут быть нелинейными. Минимизация линейной модели выполняется относительно функции качества, производная которой по параметрам приравнивается к нулю. Получающиеся нормальные уравнения решаются для определения параметров a_k.

Линейные регрессионные модели:

Линейный тип − Y = a+bx.

Квадратичный тип − Y = a+bx+cx^2.

Степенной вид − Y = a+bx+cx^2+dx^3 + ...

Программа использует метод Левенберга-Маркквардта (LM) для решения уравнений нелинейной регрессии. Этот метод комбинирует метод крутого спуска и ряд Тейлора. Первый из них работает лучше вдали от минимума, а ряд Тейлора – наоборот – вблизи минимума. LM-алгоритм обеспечивает гладкие переходы между этими двумя методами с помощью итерационного процесса. В качестве критерия используется функция хи-квадрат, которую можно дифференцировать так, чтобы получить вектор градиента и матрицу Якоби. LM-алгоритм имеет следующие шаги:

1. Вычисляют сумму квадратов невязок $\chi^2(a)$.
2. Выбирают некоторое значение для параметра λ (0,001 в CurveExpert).
3. Решают линейные уравнения относительно δa_l:

$$\sum_{k=1}^{Na} M_{kl} \cdot \delta a_l = G_k, \text{ где } M_{ii} = C_{ii}(1+\lambda), \quad M_{ij} = C_{ij} \text{ для } i \neq j,$$

$$G_k = -\frac{1}{2}\frac{\partial \chi^2(a)}{\partial a_k} - \text{градиент}, \quad C_{ij} = \frac{\partial^2 \chi^2(a)}{\partial a_i \partial a_j} - \text{элемент матрицы Гессе}.$$

4. Оценивают $\chi^2(a+\delta a)$.
5. Если $\chi^2(a+\delta a) > \chi^2(a)$, то увеличивают λ в κ раз (κ=10 в CurveExpert), и процесс вычислений переходит на шаг 3.
6. Если $\chi^2(a+\delta a) < \chi^2(a)$, то уменьшают λ в κ раз, корректируют вектор параметра $a = a + \delta a$, и процесс вычислений переходит на шаг 3.

Итерации заканчиваются, когда: $|\chi^2(a+\delta a) - \chi^2(a)| < \varepsilon$ (точность). Эта точность указывается в пункте File|Preferences.

В CurveExpert нелинейные модели был разделены на семейства, основанные на их характерном поведении. Эти семейства и их члены перечислены ниже.

Показательные модели имеют показательные или логарифмические базовые функции. Они вообще выпуклые или вогнутые кривые, но некоторые модели в этой группе могут иметь точку перегиба и максимум или минимум:

Показательный	Y=a*exp (b*x)
Измененный показательный	Y = a*exp (b/x)
Логарифм	Y = a+b*ln (x)
Обратный логарифм	Y = 1 / (a+b*ln (x))
Модель давления пара	Y = exp (a+b/x+c*ln (x))

Степенные модели - набор выпуклых или вогнутых кривых без точек перегиба или максимумов / минимумов:

Степенной	Y = a*x^b
Модифицированный степенной	Y = a*b^x

Смещенный степеннной	Y = a*(x-b) ^c
Геометрический	Y = a*x ^ (b*x)
Измененный геометрический	Y = a*x ^ (b/x)
Корень	Y = a^ (1/x)
Модель Hoerl	Y = a*(b^x) * (x^c)
Измененная модель Hoerl	Y = a*b ^ (1/x) * (x^c)

Модели плотности урожая широко используются, особенно в сельскохозяйственных приложениях:

Взаимная модель	Y = 1 / (a+ bx)
Взаимная квадратичная	Y = 1 / (a+ bx + cx^2)
Bleasdale модель	Y = (a+ bx) ^ (-1/c)
Harris модель	Y = 1 / (a+ bx^c)

Модели роста характеризуются монотонным ростом от некоторой установленной величины до асимптоты. Эти модели часто используются в технике.

Показательный (2)	Y = a* (1-exp (-bx))
Показательный (3)	Y = a* (b-exp (-cx))
Рост насыщенности	Y = ax / (b + x)

Унимодальное семейство представляет «S-форменные» кривые роста в биологии, сельском хозяйстве и экономике. Эти кривые начинаются в заданной точке и увеличиваются монотонно до точки перегиба. После этого они приближаются асимптотически к некоторому уровню. Это семейство - фактически поднабор семейства роста, но отделено в CurveExpert из-за их особого поведения:

Gompertz модель	Y = a* exp (-exp (b - cx))
Логистическая модель	Y =a / (1 + exp (b - cx))
Richards модель	Y =a / (1 + exp (b - cx)) ^ (1/d)
MMF модель	Y = (ab + cx^d) / (b + x^d)
Weibull модель	Y =a - b*exp (-cx^d)

Пёстрое семейство. Некоторые данные не вписываются в упомянутые модели;

Синусоидальная	Y = a+ b*cos (c*x + d)
Гауссова модель	Y = a*exp ((- (x - b) ^2) / (2*c^2))
Гиперболическая	Y = a+ b/x
Модель высокой температуры	Y = a+ bx + c/x^2
Рациональная функция	Y = (a+ bx) / (1 + cx + dx^2)

CurveExpert позволяет определять собственные нелинейные модели регресса. Параметры должны начаться с '@' и записываться последовательно (до 19 параметров). Все арифметические действия имеют силу, также как символ ^. Имеется пример диалога:

A*x*log (b+x) +c^x.

Вводится только правая часть. Поддерживаются логические операторы:

(A+b*x) * (x < =5) + (c+d*x) * (x > 5).

2.2. Домашнее задание

1. Получить от преподавателя задание на разработку одного из m-файлов проекта «Параметрические модели БНЭ».
2. На основе приведенных в п. 2.1 математических соотношений и примеров программирования составить заготовку заданного m-файла.
3. Предложить иные, нежели в п. 2.1, алгоритмы разработки проекта «Параметрические модели БНЭ», в том числе с помощью программы CurveExpert.
4. Подготовить набор данных ВАХ диода и ответы на контрольные вопросы.

2.3. Лабораторное задание

1. Запустить систему MATLAB (версия 6.x), выбрать в главном меню команду **File / New / M-file** и в открывшемся окне редактора ввести текст заготовки m-файла.
2. Провести отладку и обустройство (вставить help-текст, комментарии к каждой команде) m-файла. Согласовать обозначение глобальных параметров, количество и типы входных и выходных переменных с программами-«соседями». Окончательный результат отладки m-файла показать преподавателю.
3. Проверить работу m-файла в автономном режиме:
   - составить набор типовых данных входной информации;
   - проверить рабочее пространство на наличие входных данных и глобальных параметров (команда **File / Show Workspace**);
   - исправить появившиеся ошибки исполнения программы.
4. Включить полностью отлаженный m-файл в проект «Параметрическая модель БНЭ». Провести с другими студентами отладку совместной работы программ проекта:
   - исключить ошибки выполнения во всех режимах работы алгоритма;
   - подобрать форму ВАХ и параметры гармонического сигнала для наиболее представительного варианта расчёта спектра и запомнить их;
   - сохранить результаты отладки на гибком диске.
5. Скопировать несколько наиболее интересных окон расчёта параметров моделей аппроксимации через буфер обмена Windows в текстовый процессор Microsoft Word и показать их преподавателю. При наличии замечаний исправить их, не стирая с жесткого диска старые версии программ проекта и копий окон в текстовом процессоре.
6. Запустить программу CurveExpert и провести оценку параметров ВАХ полупроводникового диода при полиномиальной и показательной аппроксимациях. Сравнить значения параметров, полученные с помощью системы моделирования MATLAB, и программы CurveExpert. В случае их

существенного отличия (более 10 %) найти источник различия и повторить процесс параметрического оценивания.
7. Найти с помощью программы CurveExpert тип модели, дающий наименьшую ошибку аппроксимации. Записать её аналитическое выражение, параметры и величину ошибки. Сравнить с ошибками при полиномиальной и показательной аппроксимациях.

2.4. Содержание отчёта

1. Цель работы.
2. Структурная схема алгоритма проекта «Параметрические модели БНЭ» и её выделенный фрагмент, являющийся индивидуальным заданием на разработку m-файла.
3. Описание алгоритма (математическая модель, формулы, рекуррентные соотношения при символьном описании) индивидуального задания.
4. Отлаженная версия разработанного m-файла с полным набором комментариев.
5. Состав входной и выходной информации и копии окон с графиками при трёх типах аппроксимации: кусочно-линейной, степенной и экспоненциальной. Параметры моделей.
6. Результаты сравнения параметров моделей, полученных с помощью системы моделирования MATLAB и программы CurveExpert. Параметры наиболее точной модели, аппроксимирующей ВАХ диода.
7. Выводы по работе. Достоинства и недостатки встроенного языка системы MATLAB. Сравнение системы моделирования MATLAB и программы CurveExpert по результатам решения поставленной задачи.

2.5. Контрольные вопросы

1. Что такое глобальные параметры системы MATLAB и как посмотреть в рабочем окне их значение?
2. С помощью каких процедур можно определить минимум функции нескольких аргументов?
3. В чём состоит различие функций **ezplot**, **subplot**, **fplot** и **plot**?
4. Каким образом в одном окне можно построить несколько графиков и вывести текст в заданную область этого окна?
5. Приведите два примера выбора альтернативы при организации интерфейса в системе MATLAB?
6. С помощью каких операторов встроенного языка системы MATLAB можно организовать обработку одного, двух и нескольких условий?
7. Как выглядит типовой формат m-файла?
8. В чём состоит отличие Script-файла от m-файла?
9. Каков порядок разработки отдельных m-файлов в составе сложного проекта, состоящего из нескольких взаимосвязанных между собой файлов?

10. Какие различия могут наблюдаться, на ваш взгляд, в спектрах тока, полученных расчётным и аналитическим путями?
11. Назовите основные типы моделей, применяемые в программе CurveExpert.
12. Как получить параметры кусочно-линейной аппроксимации с помощью программы CurveExpert?
13. Поясните суть метода Левенберга-Маркквардта при вычислении параметров нелинейных моделей.

Лабораторная работа № 3

ИССЛЕДОВАНИЕ И ПРИМЕНЕНИЕ ГЕНЕРАТОРОВ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ

Цель работы − ознакомление с алгоритмами генерации случайных чисел и методами формирования на их основе числовых последовательностей с заданными вероятностными и корреляционными свойствами.

Работа состоит из четырех частей: домашнего задания, коллоквиума, расчетно-экспериментальной части и оформления полученных результатов в виде отчета. При подготовке домашнего задания следует использовать сведения, приведённые во вводной части работы [3, 4, 5].

3.1. Краткие сведения об алгоритмах статистического моделирования

3.1.1. Определение объёма выборки

Основным методом получения результатов с помощью имитационных моделей является метод статистического моделирования на ЭВМ. Число положительных исходов в статистическом эксперименте объёма N обозначим n. Тогда отношение $f_n = n/N$ называется относительной частотой выполнения заданного условия. При больших N частоты f_n в различных экспериментах приблизительно одинаковы. Существует некоторое значение p, называемое вероятностью заданного события, около которого группируются указанные частоты. В пределе при $N \to \infty$ $p = \lim n/N$, поэтому f_n называют эмпирической оценкой вероятности p.

Схема испытания Бернулли - статистический эксперимент, в каждом испытании которого наблюдается либо 1 (положительный исход), либо 0 (отрицательный исход) с вероятностями соответственно p и $q = 1-p$.

Методы Монте-Карло. Такими методами называются численные методы решения математических задач при помощи моделирования случайных величин и статистической оценки их характеристик.

Если задать некоторую область допустимых погрешностей $J \pm d$, которая называется **доверительным интегралом**, то вероятность P_d полу-

чения оценки, не выходящей за пределы этой области, может быть сделана как угодно близкой к единице при достаточно большом увеличении объёма выборки N. Очевидно,

$$P_d = \int_{J-d}^{J+d} f(x)dx,$$

где $f(x)$ – плотность распределения; d – квантили распределения для доверительной вероятности P_d. Таким образом, важнейший прием построения методов Монте-Карло есть сведение задачи к вычислению выборочных средних. Для этого необходимо:

1) выбрать, исходя из предварительных сведений о задаче, наиболее удобный вид распределения случайной величины X;

2) уметь находить выборочные значения $x1, x2,$ случайной величины X;

3) правильно определять необходимый объём выборки N.

Ответ на первый вопрос находится в результате анализа постановки задачи, исходя из имеющихся возможностей моделирования случайных величин и опыта работы с вероятностными моделями. Во многом этот ответ носит творческий характер и является следствием компромиссов. В подавляющем большинстве вероятностные модели базируются на двух случайных величинах: $R(0,1)$ - равномерно распределенной на интервале [0,1] и $N(0,1)$ - нормально распределенной с нулевым математическим ожиданием и единичной дисперсией.

Наиболее легко получить ответ на третий вопрос в схеме испытаний Бернулли. В этом случае оценка p' математического ожидания подчиняется биномиальному распределению, имеющему следующие характеристики: математическое ожидание p и дисперсия $\sigma^2 = p(1-p)/N$ при объёме выборки N. Вероятность отклонения p' от его математического ожидания подчиняется неравенству Чебышева:

$$Pr(|p' - p| > d) < \sigma^2/(d \cdot d).$$

Слева стоит вероятность выхода оценки за пределы доверительного интервала d. Она равна $Qd = 1-Pd$, где Pd – доверительная вероятность. Объём выборки N должен удовлетворять, таким образом, соотношению

$$N > p(1-p)/(d \cdot d \cdot Qd).$$

Это выражение нельзя использовать для вычисления N, так как p здесь является искомой (неизвестной) величиной. Поскольку максимальное значение числителя достигается при p=0,5, то для N можно записать усиленное неравенство, более грубое, но пригодное для оценок:

$$N = 0{,}25/(d \cdot d \cdot Qd).$$

Анализ этого выражения показывает, что методы Монте-Карло не годятся для получения решений с высокой точностью. Например, для вычисления выборочного среднего с точностью d=±0,01 и доверительной вероятностью $Pd > 0{,}99$ необходимо выполнить:

$$N > 0{,}25/(0{,}001 \cdot 0{,}001 \cdot (1-0{,}99)) = 25e6,$$

т.е. 25 миллионов испытаний. Такой объём выборки даже при решении простых задач трудно выполнить на современных ЭВМ. Например, при реализации каждого испытания за 1 мс на весь эксперимент - определение одного числа (выборочного среднего) - необходимо затратить около 8 часов чистого машинного времени. Однако в практических расчетах такие точности не требуются, и применение методов Монте-Карло не вызывает серьезных трудностей.

При большом числе испытаний (N>100) распределение $f(p)$ эмпирической оценки вероятности p близко к нормальному распределению, поэтому для вычисления значений N можно использовать уточненное расчетное соотношение:

$$N = 0{,}25 * t_p * t_p / (d*d),$$

где t_p - квантиль нормального распределения, соответствующий доверительной вероятности P_d и определяемый по функции Лапласа:

$$t_p = \Phi^{-1}((1+P_d)/2).$$

Поскольку $t_p * t_p < 1/Q_d$, то избыточность объёма выборки уменьшается. Например, при P_d = 0,68 квантиль t_p = 1, поэтому число испытаний по уточненной формуле примерно в три раза (1/0.32) меньше, чем по грубой формуле. Избыточность в значении N, вызванная заменой выражения $p(1-p)$ его максимальным значением 0,25 рекомендуется уменьшить, проводя серию из M экспериментов по $N1$ испытаний ($N1 = N/M$). После каждой очередной m-й серии проверяется условие

$$\frac{d^2}{t_{m-1,\,p}{}^2} < \frac{\sigma^2}{m-1} = \frac{1/m\sum(a_k - a_{Nm})^2}{m-1},$$

где $a_k = n_k/N1$ - относительная частота «успехов» в k-й серии;

$a_{Nm} = \frac{1}{m}\sum_{k=1}^{m} a_k$ - то же во всех m сериях;

$t_{m-1,p}$ - квантиль распределения Стьюдента с m степенями свободы. Он находится с учетом двустороннего доверительного интервала. Если условие не выполняется, то проводится очередная серия из $N1$ испытаний. В противном случае объём выборки считается достаточным, а величина a_{Nm} - удовлетворительной оценкой вероятности.

Пример 1. Вычислим методом Монте-Карло оценку некоторой вероятности p с доверительным интервалом $d = \pm 0{,}01$ и доверительной вероятностью $P_d = 0{,}9$. Принимая, что оценка распределена по нормальному закону (тогда $tp = 1.645$), определим грубо объём выборки $N > 0{,}25*1{,}65*1{,}65 /(0{,}001*0{,}001) = 6800$. Расчёт проводится сериями по 500 испытаний. Тогда количество серий не превышает 14. Квантили распределения Стьюдента при $P_d = 0{,}9$:

m	2	3	4	5	6	7	8	9	10	12	14
t_{m-1}	2,92	2,35	2,13	2,02	1,94	1,89	1,86	1,83	1,81	1,78	1,76

После выполнения расчетов по двум сериям испытаний выборочная дисперсия должна быть меньше $d*d/(t_1*t_1)$ = 0,01*0,01/(2,92*2,92) = 1,2E-05. При невыполнении этого условия вычисления продолжаются.

Часто оценку математического ожидания необходимо получить на интервале, отличном от единичного, или тогда, когда случайная величина принимает не два, а множество значений. В этих случаях дисперсия оценки может быть любой конечной величиной, поэтому пользоваться приведенной выше формулой для оценки объёма выборки нельзя. Однако величину дисперсии нетрудно оценить эмпирически, в ходе расчетов. Для этого достаточно одновременно с вычислением суммы случайных величин sum(X) накапливать также сумму их квадратов sum(X^2). При больших N (более 100) выборочная дисперсия определяется по формуле

$$\sigma^2 = \text{sum}(X^2)/(N-1) - (\text{sum}(X)/N)^2 .$$

Это соотношение можно использовать в критерии окончания процесса вычислений при подстановке σ^2 вместо константы 0,25.

3.1.2. Получение равномерно распределенных случайных чисел

При формировании вектора случайных параметров используют случайные числа с различными законами распределения. В курсе теории вероятностей показывается, что двух видов случайных величин - равномерно $R(0,1)$ и нормально $N(0,1)$ распределенных - достаточно для получения из них случайных величин большинства распределений, встречающихся в практике моделирования. Более того, от двух равномерно распределенных случайных величин можно перейти к двум нормальным. Поэтому в качестве исходной обычно выбирают непрерывную случайную величину r, равномерно распределенную на интервале (0,1). Ее математическое ожидание Mr = 0,5, дисперсия Dr = 1/12, плотность $fr(x)$ = 1 и функция распределения $Fr(x) = x$ при x, принадлежащем интервалу (0,1).

Представим случайное число в виде бесконечной десятичной дроби

$$r = 0, x1\ x2\ x2\ \ldots\ xi\ \ldots\ ,$$

где десятичные цифры, обозначаемые $x1, x2, \ldots, xi, \ldots$, очевидно, также случайны. Доказано, что *если эти цифры независимы друг от друга и могут принимать с одинаковой вероятностью* 1/10 *одно из* 10 *значений* {0, 1, 2, ... 9}*, то число r равномерно распределено на интервале (*0,1*)*. Верно также обратное утверждение. Эта теорема справедлива для любой формы представления числа, необязательно десятичной. Наибольший интерес имеет двоичная форма, когда цифры xi могут принимать только два значения 0 и 1 с равной вероятностью $p(0) = p(1) = 0,5$. Следовательно, если на машине организовать простую схему независимых испытаний Бернулли с вероятностью «успеха» (обозначаемого теперь как 1), равного 0,5, то при бесконечно большом объёме выборки и представлении результатов испытаний в форме 0. $x1\ x2\ x2$ ... можно получить равномерно распределенное случайное число r. Поскольку в вычислениях всегда используют числа с

конечным количеством знаков, то схема Бернулли должна прерываться на некотором *k*-м испытании. Тогда случайное число будет представлять собой конечную двоичную дробь. Отметим, что никакого округления здесь не производится, а все двоичные цифры *xi* статистически однородны. В реальных ЭВМ величина *k* определяется количеством разрядов, отводимых на мантиссу (от 22 до 48).

Метод псевдослучайных чисел - наиболее широко распространенный метод получения случайных чисел на ЭВМ. Суть его состоит в том, что машина по заранее заданному алгоритму определяет случайное число. При M разрядах мантиссы в ЭВМ может быть сформировано ровно 2^M Чисел (в частности, при $M = 20$ количество их превышает миллион). Поэтому основным требованиям к алгоритму является «отсутствие», детерминированности, полная независимость в процессе выбора очередного числа среди 2^M чисел. Противоречие между этим требованием и абсолютной детерминированностью любого вычислительного процесса, выполняемого ЭВМ, снимается с помощью специальных алгоритмов, проверенных поколениями программистов.

Большинство таких алгоритмов представляют собой рекуррентные формулы первого порядка:

$$r_{i+1} = A(r_i),$$

где начальное значение r_0 должно быть задано. Для получения «хорошей» последовательности $r_1, r_2, r_3, \ldots$ псевдослучайных чисел необходимо, чтобы функция $y = A(x)$ имела график, плотно заполняющий единичный квадрат. Наиболее часто за такую функцию выбирают взятие дробной части от произведения большого числа M и аргумента:

$$r_{i+1} = M * r_i - \text{entier}(M * r_i / L) * L = (M * r_i) \bmod L,$$

где entier(*) - функция взятия целой части. Если число M достаточно велико (более нескольких тысяч), то пары соседних чисел (r_1, r_2), (r_3, r_4), ... дают координаты точек, практически равномерно распределенных в единичном квадрате. Вследствие неоднозначности связи между аргументом и функцией эти точки выпадают почти независимо друг от друга, поэтому рекуррентное соотношение можно использовать для получения псевдослучайных чисел. При этом цифры двоичной дроби образуют почти случайную последовательность испытаний Бернулли.

Важнейшими характеристиками алгоритмов такого вида являются **длина отрезка апериодичности** P и **длина периода** T. Первая представляет собой количество псевдослучайных чисел от начала последовательности до первого числа, которое совпадает с каким-либо из предыдущих. Далее оно повторяется с периодом T, следовательно, последовательность после P-го числа становится периодической. Отрезок апериодичности состоит из различных чисел, его средняя длина может быть оценена по формуле $P = \sqrt{(\pi * N / 2)}$, где в качестве N выбирается количество возможных значений функции $A(x)$ в конкретной ЭВМ. В частности, при длине

мантиссы в M разрядов $N=2^M$. В статистических расчетах необходимо использовать не более чем P чисел последовательности. Заметим, что такая оценка длины апериодичности очень «осторожна», на самом деле величина P значительно больше.

Наибольшее распространение получил алгоритм, предложенный Д. Лемером. Он называется методом сравнений, его формула соответствует вышеприведённой формуле с точностью до постоянного множителя. Этот множитель введен для того, чтобы выполнять все промежуточные вычисления с целыми числами и только окончательный результат приводить к интервалу (0,1). Очевидно, таким множителем может быть любое число, кратное 2^M.

Пример 2. Рассмотрим алгоритм формирования равномерно распределенных псевдослучайных чисел на БЭСМ-6 (язык программирования Фортран).

```
FUNCTION RANDOM(I)
I1 = I*3125
I0 = I1/67108864
I = I - I0*67108864
RANDOM = I/67108864
RETURN
```

Здесь вычисления проводятся по формуле $I_{i+1} = (I_i * M) \bmod L$, где все переменные целые: $M=3125=5^5$, $L=67108864=2^{26}$. Перед первым обращением к функции её аргументу - целой переменной I - присваивается любое нечетное число в диапазоне от 1 до 67108863. На получение одного псевдослучайного числа затрачивается два умножения и одно деление целых чисел, одно деление целого на вещественное.

Средняя длина апериодичности при $N = L$ равна примерно 10000. В случае, когда выбраны $M = 5^{(2*k+1)}$, k=0,1,2,...; $L = 2^m$, $m > 2$, длина отрезка апериодичности может быть определена по точной формуле $P = 2^{(2*m-2)} = L/4$. Поскольку здесь $M=5^5$ ($k = 2$), $L = 2^{26}$ ($m = 26$), то $P = 16777216 = 2^{24}$. Конечно, в абсолютном большинстве случаев статистического моделирования такой отрезок апериодичности псевдослучайной последовательности вполне достаточен.

Пример 3. Датчик равномерно распределенных псевдослучайных чисел, входящий в библиотеку стандартных подпрограмм IBM PC имеет тот же алгоритм, что в примере 2, но в нем применены другие константы: $M = 5^{13} = 1220703125$; $L = 2^{31} = 2147483648$. Число $m = 31$ выбрано здесь потому, что на целое число в стандартной форме отводится 31 двоичный разряд, следовательно, при формировании псевдослучайной последовательности используется вся длина регистра АУ, включая его младшие разряды. В этом случае увеличивается длина отрезка апериодичности, но несколько замедляется работа датчика. Как и в предыдущем примере, в качестве начального значения рекуррентного процесса выбирается любое нечетное число в диапазоне от 1 до 2^{31}-1.

Моделирование нормально распределенных параметров компонентов также осуществляется с помощью датчиков равномерно распределенных псевдослучайных чисел (РРЧ). Существуют много методов перехода от датчиков РРЧ к датчикам нормально распределенных чисел (НРЧ), однако в широкой вычислительной практике нашли применение только два.

Метод конечной суммы (суперпозиции). Пусть имеется n независимых равномерно распределенных величин r_i. Если образовать из них сумму:

$$s(n) = \sqrt{3/n} \cdot \sum_{i=1}^{n} (2r_i - 1),$$

то, согласно центральной предельной теореме, при $n \to \infty$ распределение случайной величины $s(n)$ стремится к нормальному с нулевым математическим ожиданием и единичной дисперсией:

$$s(n) \to N(0,1) \leftrightarrow f(x) = \exp(-x^2/2) / \sqrt{2\pi}.$$

Асимптотика устанавливается весьма быстро: уже при $n = 6$ отличия от распределения $N(0,1)$ заметны только при больших значениях аргумента (s>2). На практике обычно выбирают $n = 12$, что, помимо увеличения точности моделирования, позволяет увеличить его скорость (нормирующие множители в этом случае равны единице).

Метод аналитических преобразований. Пусть r_1 и r_2 - два независимых РРЧ. Тогда два случайных числа, представляющих аналитические однозначные функции от r_1 и r_2, можно также интерпретировать как полярные координаты $p = -\ln(r_2)$, $\phi = 2\pi * r_2$ некоторой точки, случайным образом распределенной на всей плоскости. Случайные величины p и ϕ, очевидно, взаимно независимы, при этом радиус-вектор p распределен по экспоненциальному закону, а его угол ϕ есть РРЧ на интервале $(0, 2\pi)$. Совместное распределение $w(p, \phi)$ есть произведение распределений $w(p)$ и $w(\phi)$, равное $\exp(-p)/(2\pi)$. В то же время декартовы координаты случайных точек также взаимно независимы. Они распределены по нормальному закону $N(0,1)$, их совместное распределение будет иметь вид:

$$w(x, y) = w(x)*w(y) = \exp((-x^2 - y^2)/2)/c = \exp(-r^2/2)/c,$$

где $c = 2\pi$. Таким образом, проведя указанные выше аналитические преобразования над двумя РРЧ и перейдя к декартовым координатам, можно получить сразу два независимых НРЧ:

$$s1 = \sqrt{(-2*\ln(r_1))}*\cos(2\pi * r2), \quad s2 = \sqrt{2}\,(-2*\ln(r_2)*\sin(2\pi * r2).$$

Заметим, что количество точек, попавших в кольцо $(r, r+dr)$, распределено по релеевскому закону.

Достоинством датчика НРЧ, работающего по формулам, является отсутствие ошибок моделирования нормального распределения и одновременная генерация двух независимых НРЧ, что бывает удобно при имитации комплексных, нормально распределенных величин. Несмотря на то, что вычисления проводятся по четырем сложным функциям, все же на

большинстве ЭВМ такой датчик работает на 25 − 40 % быстрее, чем датчик, реализующий формулу суммирования.

Для моделирования случайных величин, имеющих произвольный закон распределения, используется метод обратных функций. Суть его состоит в следующем. Интегральная функция любой случайной величины имеет значения в интервале (0,1), т.е. в том же интервале, что и равномерно распределенная величина. Таким образом, приравнивая приращения вероятностей при одинаковых значениях интегральных функций равномерного $F_r(x)$ и моделируемого $F_m(x)$ распределений и разрешая полученное уравнение относительно y, можно получить явную формулу:

$$y = F_m^{-1}(x),$$

предназначенную для моделирования случайной величины с требуемым законом $F_m(x)$. В формуле $F_m^{-1}(*)$ - обратная функция по отношению к $x = F_m(y)$. В некоторых случаях уравнение приходится решать численно. Например, при получении НРЧ методом обратных функций аналитическое решение в замкнутом виде невозможно.

Пример 4. Выполним моделирование экспоненциальной случайной величины e, имеющей распределение $f_m(y) = c*\exp(-c(y - y_0))$.

Так как:

$$F_m(y) = 1 - \exp(-c(y - y_0)),\ y>y_0,$$

то необходимо решить относительно y уравнение:

$$1 - \exp(-c(y-y_0)) = x.$$

Явные выражения для расчета величины e будут иметь вид:

$$e = y_0 - \ln(1- r)/c \text{ или } e = y0 - \ln(r)/c.$$

Здесь оба выражения справедливы, так как случайные величины r и $1- r$ распределены по одинаковому – равномерному – закону. При $y_0 = 0$ экспоненциально распределенная величина генерируется просто как натуральный логарифм РРЧ со сменой знака.

Существует много других методов перехода от РРЧ к распределениям специального вида. При этом используются специальные свойства связей распределений. Например, через переход от равномерного к бета-распределению можно генерировать случайные числа из очень большого класса распределений.

3.2. Домашнее задание

1. Ознакомиться с алгоритмами генерации равномерно распределенных чисел, приведенных в п. 3.1.
2. Составить m-файл, в котором реализована рекуррентная последовательность вычисления очередного равномерно распределенного числа $R1$:

$$R1 = \text{Lemer}(R0, M, L),$$

где M – нечетное число (простая степень 3 или 5);

L – четное число (степень 2), $L >> M$.

3. Составить m-файл, в котором выходной параметр представляет собой вектор или матрицу равномерно распределенных чисел (в зависимости от количества и значений формальных параметров):

$$R = \text{Randomn}(N1, N2),$$

где $N1$ – количество строк массива R;

$N2$ – количество столбцов массива R.

4. Составить m-файл, в котором по R – вектору или матрице равномерно распределенных чисел – вычисляется соответственно вектор или матрица экспоненциально распределенных чисел:

$$E = \text{Expon}(R).$$

5. Составить m-файл, в котором по R – вектору или матрице равномерно распределенных чисел вычисляется соответственно вектор или матрица нормально распределенных чисел:

$$N = \text{Gaussn}(R).$$

6. Продумать алгоритм определения числа апериодичности датчика равномерно распределенных чисел.
7. Выбрать тип и параметры цифрового фильтра с конечной импульсной характеристикой для формирования коррелированных временных последовательностей.

3.3. Лабораторное задание

1. Создать m-файл генерации равномерно распределенного числа **Lemer**(*R*0, *M*, *L*) и проверить его работу в рабочем окне MATLAB.
2. На основе функции **Lemer** определить число апериодичности для двухтрёх значений *L* и *M*. Найти значения параметров *L* и *M*, при которых длина апериодичности максимальна.
3. Описать параметры *R*0, *L*, *M* как глобальные (ввести строку **global** R0 L M; в рабочее окно и в m-файл) и создать функцию **Randomn** (*N*1, *N*2). Получить матрицу равномерно распределённых чисел *Rm* (1:25, 1:25) и извлечь из неё вектор *Vr* (1:100). Построить график зависимости *Vr* от номера. Воспользоваться функцией **hist** для построения гистограммы значений вектора *Vr*.
4. Получить матрицу экспоненциально распределённых чисел *Em* (1:25, 1:25) и извлечь из неё вектор *Ve* (1:100). Построить график зависимости *Ve* от номера и гистограмму его значений.
5. Получить матрицу нормально распределённых чисел *Nm* (1:25, 1:25) и извлечь из неё вектор *Vn* (1:100). Построить график зависимости *Vn* от номера и гистограмму его значений.
6. Определить корреляционные свойства сгенерированных последовательностей – матриц *Rm, Em, Nm* – с помощью функции cov. Представить в трёхмерном виде полученные ковариационные матрицы *Cr*, *Ce*, *Cn*, используя функции **mesh** (линейчатый график), **surf** (график с «заливкой»). Повернуть график с «заливкой» с помощью функции **view**(2)

так, чтобы ковариационная матрица была представлена в виде элементов различного цвета. Убедиться в том, что диагональные элементы ковариационных матриц всегда положительны и больше, чем элементы соответствующих строки и столбца.

7. Определить корреляционные свойства матрицы *Nnm*, сгенерированной с помощью встроенной функции генератора нормально распределенных чисел **randn**(25, 25). Представить в трёхмерном виде матрицу *Cnm*. Сравнить ковариационные матрицы *Cn* и *Cnm* графически и по гистограммам.
8. Получить коэффициенты *Klow* цифрового НЧ-фильтра с КИХ, обратившись к функции **fir1**(*P*, *Ws*) и выбрав порядок *P* фильтра в диапазоне от 10 до 100. Проверить правильность создания фильтра по графику его АЧХ (функция **fir1** без выходного аргумента).
9. Сгенерировать коррелированную последовательность *Vc* из вектора *Vn* с помощью функции **filter**(*Klow*, 1, *Vn*). Построить совмещенные графики последовательностей *Vn* и *Vc* и их гистограмм.
10. Составить m-функцию вычисления авто- и взаимнокорреляционной функций для вектора *CoefV* = **CovVec**(*V*1, *V*2) и вычислить эти коэффициенты для векторов *Vn* и *Vc*. Построить совмещенные графики корреляционных коэффициентов. Объяснить их различие.

3.4. Содержание отчёта

1. Цель работы.
2. Описание использованных в лабораторной работе функций системы MATLAB: назначение, входные и выходные параметры, текст m-функции.
3. Длины апериодичности в зависимости от параметров *L* и *M*.
4. Графики и гистограммы, полученные в пунктах 3, 4 и 5 лабораторного задания.
5. Результаты сравнения ковариационных матриц *Cn* и *Cnm*: уровень и неравномерность диагональных элементов, относительный уровень и неравномерность «фона», структурные особенности матриц.
6. Порядок создания формирующего цифрового фильтра, его АЧХ и вектор коэффициентов в виде таблицы и графика.
7. Совмещенные графики последовательностей *Vn* и *Vc*, их гистограмм и графики корреляционных коэффициентов, полученные в пунктах 9, 10 лабораторного задания, и их объяснения.
8. Выводы по работе.

3.5. Контрольные вопросы

1. Какие требования предъявляются к датчикам псевдослучайных чисел?
2. Что такое базовый датчик псевдослучайных чисел?

3. Приведите формулы конгруентных процедур генерации псевдослучайных чисел.
4. Какие методы существуют для проверки (тестирования) качества генераторов псевдослучайных чисел?
5. Как получить случайные числа с таблично заданной функцией распределения?
6. Приведите формулы преобразования областей определения псевдослучайных чисел. Каким образом при этих преобразованиях меняются их вероятностные характеристики?
7. Какие функции в системе MATLAB служат генераторами псевдослучайных чисел?
8. Как получить из последовательности независимых псевдослучайных чисел последовательность коррелированных чисел?
9. Записать равномерно распределенное случайное число, полученное как результат 22 подбрасываний наиболее верткой однокопеечной монеты. Что считать за единицу – «герб» или «решку» - безразлично. Рекомендуется сравнить время генерации такого числа с машинным (1 мкс на ПК).

Лабораторная работа № 4

ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ЛИНЕЙНОЙ МОДЕЛИ

Цель работы − изучение методов параметрического и непараметрического оценивания временных рядов и определение параметров линейной модели по вход-выходным данным системы.

Работа состоит из четырех частей: домашнего задания, коллоквиума, расчетно-экспериментальной части и оформления полученных результатов в виде отчета. При подготовке домашнего задания следует использовать сведения, приведённые во вводной части работы [6, 7].

4.1. Краткие сведения о формах представления и методах определения параметров линейных моделей

4.1.1. Формы представления моделей

При экспериментальном определении параметров модели необходимо обеспечить:

• подбор адекватной структуры модели;

• выбор такого входного сигнала, чтобы по результатам эксперимента можно было найти оценки всех параметров модели. Наиболее просто задача определения параметров решается для *линейных* объектов, для которых выполняется принцип суперпозиции. Здесь можно выделить два случая:

1) объект линеен по входному воздействию:

$$y(t) = \Psi[\beta_1 u_1(t) + \beta_2 u_2(t)] = \Psi[\beta_1 u_1(t)] + \Psi[\beta_2 u_2(t)] = y_1(t) + y_2(t);$$

2) объект линеен по параметрам:

$$y(t) = \Psi[\beta_1 + \beta_2, u(t)] = \Psi[\beta_1, u(t)] + \Psi[\beta_2, u(t)] = y_1(t) + y_2(t).$$

В задачах идентификации под линейными объектами чаще понимают объекты, линейные по входному воздействию. Под *идентификацией динамических объектов* понимают процедуру определения структуры и параметров их математических моделей, которые при одинаковом входном сигнале объекта и модели обеспечивают близость выхода модели к выходу объекта при наличии определённого критерия качества. Обычно идентификация — многоэтапная процедура. Основные её этапы перечислены ниже.

1. *Структурная идентификация* заключается в определении структуры математической модели на основании теоретических соображений.

2. *Параметрическая идентификация* включает в себя проведение идентифицирующего эксперимента и определение оценок параметров модели по экспериментальным данным.

3. *Проверка адекватности* — проверка качества модели в смысле выбранного критерия близости выходов модели и объекта.

Рассмотрим основные виды моделей линейных непрерывных стационарных динамических объектов и их взаимосвязь (действием шума $e(t)$ пока пренебрегаем).

1. Дифференциальное уравнение. Наиболее универсальная модель, имеющая форму:

$$\sum_{i=0}^{na} a_i y^{(i)}(t) = \sum_{j=0}^{nb} b_i u^{(i)}(t),$$

где na — порядок модели $(na > nb)$; a_i и b_i — постоянные коэффициенты (параметры модели), $u^{(i)}(t)$ и $y^{(i)}(t)$ — производные соответственно входного и выходного сигналов.

2. Передаточная функция. Данная характеристика определяется как отношение преобразований Лапласа выходного и входного сигналов, что с учетом свойств данного преобразования и вышеприведенной формулы дает:

$$W(p) = \frac{L\{y(t)\}}{L\{u(t)\}} = \frac{Y(p)}{U(p)} = \frac{\sum_{j=0}^{nb} b_i p^i}{\sum_{i=0}^{na} a_i p^i},$$

где $L\{\bullet\}$ — символ преобразования Лапласа; p — комплексная переменная.

3. Импульсная характеристика (*ИХ*) $g(t)$. Под ИХ понимается реакция предварительно невозмущенного объекта (то есть объекта с нулевыми начальными условиями) на входной сигнал в виде δ-функции.

4. Переходная функция *h(t).* Это реакция предварительно невозмущенного объекта на входной сигнал в виде единичного скачка. Из теории управления известны следующие соотношения между этими характеристиками:

$$L\{g(t)\} = W(p), \quad g(t) = dh(t)/dt, \quad L\{h(t)\} = W(p)/p.$$

При нулевых начальных условиях связь между выходным и входным сигналами описывается интегралом свертки:

$$y(t) = \int_{-\infty}^{+\infty} g(t-\tau)u(\tau)d\tau$$

или в операторной форме:

$$Y(p) = W(p)U(p).$$

5. Частотные характеристики. Частотные характеристики объекта определяются его комплексным коэффициентом передачи $W(j\omega) = W(p)|_{p=j\omega}$, который является Фурье-преобразованием ИХ. Модуль комплексного коэффициента передачи $|W(j\omega)| = A(\omega)$ представляет собой, как известно, амплитудно-частотную характеристику (АЧХ) объекта с передаточной функцией *W(p),* а аргумент $\arg(W(j\omega)) = \varphi(\omega)$ - фазочастотную характеристику (ФЧХ). Графическое представление $W(j\omega)$ на комплексной плоскости при изменении частоты ω от 0 до ∞ (график амплитудно-фазовой характеристики (АФХ) в полярных координатах) в отечественной литературе называется *годографом,* а в англоязычной - *диаграммой Найквиста.* В теории управления часто используется логарифмическая амплитудно-частотная характеристика (ЛАЧХ), равная $20 \lg |W(j\omega)|$.

6. Модель переменных состояния. При выборе n координат системы в качестве переменных ее состояния (такими координатами, например, могут быть выходной сигнал $y(t)$ и $n-1$ его производных) $x_i(t)$, $i = 1, 2, ..., n$, линейную систему можно описать уравнениями для переменных состояния:

$$dX(t)/dt = AX(t) + Bu(t),$$

$$y(t) = CX(t) + Du(t),$$

где $X(t) = [x_1(t), x_2(t), ..., x_n(t)]^T$ - вектор-столбец переменных состояния; A, B, C и D при скалярных *u(t)* и $y(t)$ - соответственно матрица размера $n \times n$, векторы размера $n \times 1$ и $1 \times n$ и скаляр (при векторных $u(t)$ и $y(t)$ - матрицы соответствующих размеров). Применение при нулевых начальных условиях к последним уравнениям преобразования Лапласа позволяет получить следующее выражение для передаточной функции:

$$W(p) = C(pI - A)^{-1}B + D,$$

где I — единичная матрица.

Отметим, что все приведенные модели являются эквивалентными, то есть, зная любую из них, можно получить все остальные. Для объектов, функционирование которых по тем или иным причинам представляется для дискретного времени $t_k = k*T$ (в данном случае T — интервал дискрети-

зации), то есть для дискретных объектов наиболее общим видом описания является разностное уравнение (аналог дифференциального)

$$y_k + a_1 y_{k-1} + a_2 y_{k-2} + \ldots + a_{na} y_{k-na} = b_1 u_{k-1} + b2u_{k-2} + \ldots + b_{nb} u_{k-nb}.$$

Связь между сигналами может быть отражена также через дискретную свертку:

$$y_k = \sum_{i=0}^{k} g_i u_{k-i},$$

где g_i - координаты весовой решетчатой функции объекта, или, с использованием аппарата *Z*-преобразования

$$Y(z) = \sum_{k=0}^{\infty} y_k z^{-k},$$

где $z = e^{pT}$.

Возможны следующие способы перехода от непрерывных моделей к дискретным.

1. С применением *z*-преобразования со следующей цепочкой переходов:

$$W(p) -> L^{-1}\{W(p)\} = g(t) -> g(kT) = g_k -> W(z) = Z\{w_k\}.$$

2. С заменой разностями производных в дифференциальном уравнении, описывающем непрерывный объект:

$$\frac{dy(t)}{dt} \approx \frac{y_k - y_{k-1}}{T}, \quad \frac{d^2 y(t)}{dt^2} \approx \frac{y_k - 2y_{k-1} + y_{k-2}}{T^2}$$

и т.д. (данный подход дает приемлемую точность только при малых *T*).

3. Билинейное преобразование $p = 2/T(z - 1)/(z + 1)$ (приближенный способ, предложенный А. Тастиным).

4. Переменные состояния. Для дискретных объектов также может быть использовано описание через переменные состояния:

$$X_k = AX_k + Bu_k,$$
$$y_k = CX_k + Du_k.$$

Приведем несколько распространенных моделей дискретных объектов для временной области, учитывающих действие шума наблюдения. Здесь и ниже $e(t)$ - дискретный белый шум.

1. Модель авторегрессии AR (AutoRegression) считается самым простым описанием:

$$A(z)y(t) = e(t),$$

где $A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + + a_{na} z^{-na}$.

2. ARX-модель (AutoRegressian with external input) более сложная:

$$A(z)y(t) = B(z)u(t) + e(t),$$

или, в развернутом виде:

$$y(t) + a_1 y(t-1) + a_2 y(t-2) + \ldots + a_{na} y(t-na) =$$
$$b_1 u(t) + b_2 u(t-1) + \ldots + b_{nb} u(t-nb+1).$$

Здесь $B(z) = b_1 + b_2 z^{-1} + \ldots + b_{nb} z^{-nb+1}$.

3. ARMAX-модель (AutoRegressive-Moving Average with external input – модель АРСС - авторегрессии скользящего среднего):

$$A(z)y(t) = B(z)u(t-nk) + C(c)e(t),$$

где *nk* − величина задержки (запаздывания),

$$C(z) = 1 + c_1 z^{-1} + \ldots + c_{nc} z^{-nc} .$$

4. Модель «вход-выход» (в англоязычных источниках такая модель называется «Output-Error», то есть «выход-ошибка», сокращенно OE):

$$y(t) = \frac{B(z)}{F(z)} u(t-nk) + e(t),$$

где $F(z) = 1 + f_1 z^{-1} + \ldots + f_{nf} z^{-nf}$.

5. Модель Бокса−Дженкинса (BJ):

$$y(t) = \frac{B(z)}{F(z)} u(t-nk) + \frac{C(z)}{D(z)} e(t).$$

Полиномы *B(z), F(z), C*(*z*) определены ранее,

$$D(z) = 1 + d_1 z^{-1} + \ldots + d_{nd} z^{-nd} .$$

Данные модели можно рассматривать как частные случаи обобщенной параметрической линейной структуры:

$$A(z)\, y(t) = \frac{B(z)}{F(z)} u(t-nk) + \frac{C(z)}{D(z)} e(t).$$

При этом все они допускают расширение для многомерных объектов (имеющих несколько входов и выходов).

6. Модель переменных состояния (State Space):

$$x(t+1) = Ax(t) + Bu(t),$$
$$y(t) = Cx(t) + Du(t) + v(t),$$

где *A, B, C, D* - матрицы соответствующих размеров, *v(t)* − коррелированный шум наблюдений.

4.1.2. Методы определения параметров линейных моделей

В общем случае оценивание параметров модели заданной структуры проводится путем минимизации выбранного критерия качества модели (чаще всего среднего квадрата рассогласования выходов объекта и его постулируемой модели).

1. Оценивание параметрических моделей. Оценивание параметрических моделей (метод прогноза ошибки – Predictive Error Method, сокращенно PEM) заключается в следующем. Пусть модель исследуемого объекта имеет вид так называемой обобщенной линейной модели

$$y(t) = W(z)u(t) + v(t),$$

при этом шум $u(t)$ может быть представлен как

$$v(t) = H(z)e(t),$$

где $e(t)$ - дискретный белый шум; $H(z)$ - некоторый полином от z.

Из данных выражений следует, что

$$e(t) = H^{-1}(z)[y(t) - W(z)u(t)].$$

При выборе в качестве критерия (функции потерь) величины

$$Vn(W,H) = \sum_{i=1}^{n} e^2(t)$$

оценки коэффициентов полиномов модели могут быть найдены в результате решения следующей оптимизационной задачи (в общем случае нелинейной):

$$[\overline{W}, \overline{H}) = \arg\min \sum_{i=1}^{n} e^2(t).$$

Нахождение такого решения (различными численными методами нелинейной оптимизации), как правило, достаточно сложно и трудоемко. Заметим, что еще более сложной является подобная процедура оценивания параметров модели для переменных состояния. Однако для ряда частных моделей существуют методы оценивания более простого вида. Рассмотрим их.

2. Оценивание импульсной характеристики. Предположим, что входной сигнал исследуемого (дискретного) объекта с нулевым математическим ожиданием является дискретным белым шумом, то есть имеет автокорреляционную функцию (АКФ):

$$R_u(t) = \overline{u(t+\tau)u(t)} = \begin{cases} \lambda, \textit{ если } \tau = 0, \\ 0, \quad \textit{иначе}, \end{cases}$$

где черта означает оператор усреднения; λ - интенсивность сигнала ($\lambda > 0$), сигнал не коррелирован с шумом наблюдений. В установившемся режиме, исходя из дискретного аналога уравнения свертки, который запишем в форме получим:

$$y(t) = \sum_{i=0}^{m} g(i)u(t-i) + e(t) \ ,$$

$$R_{yu}(\tau) = \overline{y(t+\tau)u(t)} = \overline{\sum_{i=0}^{m} g(i)\cdot y(t+\tau-i)} + \overline{e(t+\tau)u(t)} \; =$$

$$= \sum_{i=0}^{m} g(i)\cdot \overline{u(t+\tau-i)u(t)} = \sum_{i=0}^{m} g(i) R_u(\tau-i)\,.$$

Но в силу принятого предположения о виде АКФ входного сигнала в сумме в правой части от нуля отлично только слагаемое, соответствующее $\tau = 1$, поэтому окончательно получаем

$$R_{yu}(\tau) = \lambda g(\tau),$$

где $R_{yu}(\tau)$ - взаимная корреляционная функция (ВКФ) выходного и входного сигналов. Отсюда приходим к оценке ИХ по экспериментальным данным:

$$\overline{g(\tau)} = \frac{1}{\lambda N}\sum_{i=1}^{N} y(t+\tau)u(t)$$

где сумма в правой части с точностью до множителя представляет собой оценку взаимной корреляционной функции сигналов $y(t)$ и $u(t)$, находимую по выборкам $\{y(t)\}$, $\{u(t)\}$, $t = 1, 2, \ldots, N$.

В случае, когда входной сигнал $u(t)$ является случайным процессом, но не белым шумом, приведенным методом оценивания $g(t)$ можно воспользоваться, если предварительно, с помощью специального формирующего фильтра $\Phi(z)$ (так называемого *обеляющего фильтра*), преобразовать (хотя бы приближенно) $u(t)$ в сигнал типа белого шума: $u_\Phi(t) = \Phi(z)u(t)$. Преобразовав таким же образом $y(t)$, можно воспользоваться (с использованием выборок $\{y_\Phi(t)\}$, $\{u_\Phi(t)\}$ приведенной выше формулой для нахождения оценки $\overline{g(t)}$. Заметим, что рассмотренная процедура относится к методам корреляционного анализа.

3. Оценивание спектров и частотных характеристик. В данном случае объект (дискретный) представляется той же моделью, что и при оценивании параметрических моделей, а входной сигнал полагается случайным процессом с нулевым математическим ожиданием, спектральной плотностью $S_u(\omega)$, некоррелированным с шумом наблюдений *e(t)*, который в данном случае имеет некоторую спектральную плотность $S_v(\omega)$. Для приведенной модели связи между спектрами сигналов описываются известными соотношениями:

$$S_y(\omega) = |W(e^{j\omega T})|^2 \; S_u(\omega) + S_v(\omega), \quad S_{yu}(\omega) = W(e^{j\omega T})S_u(\omega),$$

где $S_{yu}(\omega)$ - взаимная спектральная плотность сигналов *y(t)* и *u(t),* которые можно использовать для нахождения оценок комплексного коэффициента передачи объекта $W(e^{j\omega T})$ и спектра шума $S_v(\omega)$. Методика оценивания такова:

1) сначала по данным эксперимента определяют оценки авто- и взаимной корреляционной функций:

$$\overline{R_y(\tau)} = \frac{1}{N}\sum_{i=1}^{N} y(t+\tau)y(t),$$

$$\overline{R_u(\tau)} = \frac{1}{N}\sum_{i=1}^{N} u(t+\tau)u(t),$$

$$\overline{R_{yu}(\tau)} = \frac{1}{N}\sum_{i=1}^{N} y(t+\tau)u(t);$$

2) затем находят оценки спектральных характеристик:

$$\overline{S_u(\omega)} = \frac{1}{N}\sum_{\tau=-M}^{M} \overline{R_u(\tau)} w_M(\tau) e^{-j\omega\tau},$$

$$\overline{S_y(\omega)} = \frac{1}{N}\sum_{\tau=-M}^{M} \overline{R_y(\tau)} w_M(\tau) e^{-j\omega\tau},$$

$$\overline{S_{yu}(\omega)} = \frac{1}{N}\sum_{\tau=-M}^{M} \overline{R_{yu}(\tau)} w_M(\tau) e^{-j\omega\tau},$$

где $w_M(\tau)$.- так называемое временное окно, а M − его ширина;

3) наконец, требуемые оценки определяют по соотношениям:

$$\overline{W_N(e^{j\omega T})} = \frac{\overline{S_{yu}(j\omega)}}{\overline{S_u(j\omega)}},$$

$$\overline{S_v(\omega)} = \overline{S_y(\omega)} - \frac{\overline{|S_{yu}(j\omega)|^2}}{\overline{S_u(j\omega)}}.$$

Приведенная процедура относится к методам спектрального анализа.

В системе MATLAB используется три внутренних вида матричного представления моделей: первый (для временных моделей) - так называемый тета-формат, второй (для частотных характеристик) - частотный формат, третий - формат нулей и полюсов. Адекватность моделей устанавливается применением таких критериев, как остаточная сумма квадратов ошибки - функция потерь (Loss fcn) и так называемый теоретический информационный критерий Акейке (Akaike's Infomation Theoretic Criterion - АТС).

4.1.3. Идентификация моделей в системе MATLAB

Пакет System Identification располагает большим набором функций (команд), исполняемых из командной строки и позволяющих в принципе, решать задачи идентификации, не используя графический интерфейс. Все функции (команды) делятся на следующие группы:

- графического интерфейса: **ident** и **midprefs**;
- обработки и преобразования данных: **detrend, idfilt, idinput, idresamp**;
- отображения модели: **bodeplot, ffplot, idplot, nyqplot, present, zpplot**;
- непараметрического оценивания: **covf, cra, spa, etfe**;
- параметрического оценивания: **ar, armax, arx, bj, canstart, ivar, ivx, iv4, n4sid, oe**;
- итерационного параметрического оценивания: **rarmax, rarx, rbj, roe, rpem, rplr, segment**;
- задания структуры модели: **arx2th, canform, mf2th, modstruc, ms2th, poly2th**;
- изменения и уточнения структуры модели: **fixpar, sett, thinit, unfixpar**;
- выбора структуры модели: **arxstruc, ivstruc, selstruc**;
- преобразования модели: **idmodred, th2arx, th2ff, th2par, th2poly, th2ss, th2tf, th2zp, thc2thd, thd2thc**;
- извлечения информации о модели: **getmfth, getmcap, getff, gett, getzp, th2par**;
- проверки адекватности модели: **compare, idsim, pe, predict, resid**;
- прочие: **auxvar, freqfunc, idsimd, nuderst, theta;**
- демонстрации возможностей пакета.

1. Функции непараметрического оценивания. Функция R = **covf**(z, M) выполняет расчет авто- и взаимных корреляционных функций совокупности экспериментальных данных, где z - матрица данных размером $N \times nz$, каждый столбец которой соответствует входному или выходному сигналу (обычно $z = [y\ u]$); M - максимальная величина дискретного аргумента, для которой рассчитываются корреляционные функции, минус единица. Возвращаемая величина — матрица R размером $nz \times M$.

Функция [*ir, R, cl*] = **cra**(*z, M, na, plot*) определяет оценку ИХ методом корреляционного анализа для одномерного (один вход - один выход) объекта. Здесь z - матрица экспериментальных данных вида $z = [y\ \ u]$, где y - вектор-столбец, соответствующий выходным данным; u - вектор-столбец входных данных; M - максимальное значение дискретного аргумента, для которого производится расчет оценки ИХ, по умолчанию $M = 20$; *na* - порядок модели авторегрессии (степени многочлена $A(z)$), которая используется для расчета параметров «обеляющего» фильтра $\Phi(z)$,

по умолчанию *na* = 10. При *na* = 0 в качестве идентифицирующего используется непреобразованный входной сигнал; *plot*=0 означает отсутствие графика; *plot*=1 (по умолчанию) - график полученной оценки ИХ вместе с 99 % доверительным коридором; *plot*=2 - выводятся графики всех корреляционных функций. Возвращаемые величины: *ir* - оценка ИХ (вектор значений) с 99 % доверительным интервалом для оценки ИХ; *R* - матрица, элементы первого столбца которой - значения дискретного аргумента, элементы второго столбца - значения оценки автокорреляционной функции выходного сигнала (возможно, отфильтрованного), элементы третьего столбца - значения оценки автокорреляционной функции входного сигнала (возможно, «обеленного»), элементы четвертого столбца - значения оценки взаимной корреляционной функции.

Функция [*g, phiv, z_spe*] = **spa**(*z, M, w, maxsize, T*) возвращает частотные характеристики одномерного объекта и оценки спектральных плотностей его сигналов для обобщенной линейной модели объекта (возвращая модель объекта в так называемом частотном формате): *z* - матрица исходных данных - как в рассмотренных выше функциях; *M* - ширина временного окна по умолчанию.

M = min(30, **length**(*z*)/10), где **length**(*z*) - число строк матрицы *z;* *w* - вектор частот, для которых производится расчет ЧХ, по умолчанию [1:128] / 128**pi* / *T*; *T* - интервал дискретизации; *maxsize* - параметр, определяющий максимальный размер матриц, создаваемых в процессе вычислений (оптимальный выбор этого значения позволяет добиться максимальной скорости расчетов).

Возвращаемые величины: *g* - оценка $W(e^{j\omega T})$ в частотном формате; *phiv* - оценка спектральной плотности шума *v(t); z_spe* - матрица спектральных плотностей входного и выходного сигналов.

Пример 1. Исходные данные содержатся в файле dannye.mat. Воспользуемся функцией **spa** для нахождения оценок амплитудно- и фазочастотных характеристик объекта с выводом результата в форме графиков.

```
» %3aгрузка данных
» load dannye
» z=[y  u]:
» g=spa(z);                  % Оценивание модели
» bodeplot(g)                % Построение диаграммы Боде
» w= logspace(-2, pi, 128); % Логарифмическая шкала частот (новая)
» [g, phiv] = spa(z, [  ], w); % Пустая матрица означает значение по умолчанию
» bodeplot( [ g  phiv], 3)
```

Сравнение графиков показывает, что более гладкими выглядят оценки частотных характеристик, полученные с помощью функции **spa**, чем с помощью функции **etfe**.

2. Функции параметрического оценивания. Функция [*th, refl*] = **ar** (*y, n, approach, win, maxsize, T*) оценивает параметры модели авторегрессии (АР), то есть коэффициенты полинома *A(z)* при моделировании скалярных временных последовательностей, где *y* - вектор-столбец данных, содержащий *N* элементов; *n* - порядок модели (число оцениваемых коэффициентов); аргумент **approach** (строковая переменная) определяет метод оценивания:

'*fb* ' - прямой-обратный метод (разновидность метода наименьших квадратов). Используется по умолчанию;

'*ls* ' - метод наименьших квадратов (МНК);

'*yw* ' - метод Юла-Уокера;

'*burg* ' - метод Бэрга (комбинация МНК с минимизацией гармонического среднего);

' *gl*' - метод с использованием геометрического среднего.

Если любое из данных значений заканчивается нулем (например, '*burg*0'), то вычисление сопровождается оцениванием корреляционных функций; аргумент *win* (строковая переменная) используется в случае отсутствия части данных:

win = 'now' - используются только имеющиеся данные (используется по умолчанию, за исключением случая *approach = 'yw'*);

win = 'prw' - отсутствующие начальные данные заменяются нулями, так что суммирование начинается с нулевого момента времени;

win = 'pow' - последующие отсутствующие данные заменяются нулями, так что суммирование расширяется до момента времени *N+n*;

win = 'ppw' - и начальные и последующие отсутствующие данные заменяются нулями. Используется в алгоритме Юла-Уокера.

Аргумент **maxsize** определяет максимальную размерность задачи; *T* − интервал дискретизации. Возвращаемые величины: *th* - информация о модели в так называемом тета-формате (внутреннем матричном формате представления параметрических моделей); *refl* - информация о коэффициентах и функции потерь.

Пример 2. Оценивание модели авторегрессии 4-го порядка.

```
» load dannye                    %Загрузка данных
» y1 = [ y(1:300)];
» th = ar(y, 4);              % Оценивание АР-модели
» present( th)                % Вывод информации о модели
A =
1,0000      -2,1716     1,5697          -0,3183     -0,0790
      0      0,0579          0,1379          0,1384
 0,0583
```

Матрица *A* содержит коэффициенты полинома *A(z)* (первая строка) и их стандартные отклонения (вторая строка).

Пример 3. Оценивание модели по функции **armax**.

```
» load dannye                %3агрузка данных
» z = [ y  u)];
» th = armax( z, [ 2 2 1]);          % Оценивание АРСС-модели
» present( th)                % Вывод информации о модели
B =
0         0,0073              0,0451
0         0,0021              0,0025
A =
1,0000 -1,6490       0,7026
       0  0,0097       0,0088
C =
1,0000 -0,2741       -0,4759
       0 0,0289       0,0285
```

В данном случае задана структура модели, все полиномы которой имеют 2-й порядок.

4.2. Домашнее задание

1. Ознакомиться с формами представления линейных моделей, приведенных в п. 4.1.1.
2. Составить m-файл для вычисления автокорреляционной функции временной последовательности *Ry* = **CorrY**(*Y, M*), где *Y* – временная последовательность; *M* – максимальная величина задержки, *M* < **length**(*Y*) / 2.
3. Составить m-файл для вычисления взаимной ковариационной функции временных последовательностей *Ryu* = **CovYU**([*Y U*], *M*).
4. Предложить последовательность обращения к функциям системы MATLAB, упомянутым в п. 4.1.3, для определения параметров входного и выходного сигналов.
5. Просмотреть демонстрационный файл по идентификации линейной системы (приложение System Identification системы MATLAB v.6.*).

4.3. Лабораторное задание

1. По составленным в домашнем задании m-файлам вычислить авто- и взаимные корреляционные функции входного *W* и выходного *Y* сигналов, предварительно удалив тренды командой **detrend**. Сравнить эти функции с полученными с помощью команды **covf.** Добиться графического совпадения, записать итоговые результаты в массивы *Rw, Ry, Ryw, Rwy*.
2. Вычислить и построить графики Боде собственных и взаимных спектров входного *W* и выходного *Y* сигналов. Сравнить с графиками, по-

строенными относительно линейной шкалы частот. Сохранить спектры в массивах *Sw, Sy, Syw, Swy*.

3. Найти по команде **ivar** параметры полиномов числителя и знаменателя входного *W* и выходного *Y* сигналов. Сравнить их с приведенными в файле задания коэффициентами. В случае большого различия изменить параметры команды **ivar** и повторить вычисления. Сохранить найденные коэффициенты в массивах *Ad*1 и *Bd*1.
4. Определить параметры линейной системы в методе переменных состояния по команде **canstart.** Сравнить их с приведенными в файле задания коэффициентами. В случае несовпадения изменить параметры команды и повторить вычисления. Сохранить найденные коэффициенты в массивах *Adss* и *Bdss*.
5. Пропустить входной сигнал *W* через линейные системы, описываемые полиномами (*Ad*1*, Bd*1) и (*Adss, Bdss*), используя команду **filter**. Сравнить графически выходные сигналы, объяснить различия в графиках.
6. Построить систему нулей и полюсов передаточной функции по команде **zpplot**. Пояснить форму и размеры доверительных областей вокруг нулей и полюсов.
7. Просмотреть один из вариантов демонстрационного файла по идентификации линейных систем и скорректировать его под свои данные для получения параметров модели. Сравнить с результатами, полученными в п. 3-5.

4.4. Содержание отчёта

1. Цель работы.
2. Описание составленных в лабораторной работе функций системы MATLAB: назначение, входные и выходные параметры, текст m-функции.
3. Алгоритм определения параметров в виде последовательности команд MATLAB.
4. Коэффициенты полиномов, доверительные интервалы, графики корреляционных и ковариационных функций, спектров и входного и выходного сигналов: исходные, прошедшие через модель.
5. Результаты сравнения моделей, полученных в п. 3, 4, 7 лабораторного задания.
6. График расположения нулей и полюсов модели, наилучшим образом аппроксимирующей выходной процесс при заданном входном сигнале. Коэффициенты полиномов и их доверительные интервалы этой модели.
7. Выводы по работе.

4.5. Контрольные вопросы

1. Какие формы представления линейных моделей вы знаете?
2. Приведите признаки устойчивости линейных моделей как дискретных, так и непрерывных.

3. Каким образом в формах описания линейных моделей проявляется их порядок?
4. Дайте определение идентификации динамической системы и назовите её основные этапы.
5. Какие динамические процессы можно представить моделями авторегрессии?
6. Какую роль играют автокорреляционная функция и энергетический спектр в идентификации линейных систем?
7. Что такое тренд и почему его нужно устранить перед идентификацией линейной системы?
8. Приведите алгоритм определения импульсной характеристики системы по её входному и выходному временным процессам.
9. Какие функции системы MATLAB используются для параметрического оценивания линейных динамических систем?
10. Приведите последовательность обращения к функциям системы MATLAB для получения нулей и полюсов передаточной функции системы по её входному и выходному временным процессам.

Лабораторная работа № 5

МОДЕЛИРОВАНИЕ КОНЕЧНЫХ АВТОМАТОВ

Цель работы − ознакомление с методами описания и моделирования поведения конечных автоматов на примерах детерминированного (автомат Мура) и стохастического (марковская односвязная цепь).

Работа состоит из четырех частей: домашнего задания, коллоквиума, расчетно-экспериментальной части и оформления полученных результатов в виде отчета. При подготовке домашнего задания следует использовать сведения, приведённые во вводной части работы [7, 8].

5.1. Конечные автоматы и методы их программной реализации

5.11. Дискретно-детерминированные модели

В теории автоматов (раздел теоретической кибернетики) система представляется в виде автомата, перерабатывающего дискретную информацию и меняющего свои внутренние состояния лишь в допустимые моменты времени. Автомат можно представить как некоторое устройство (черный ящик), на которое подаются входные сигналы и снимаются выходные и которое может иметь некоторые внутренние состояния. *Конечным автоматом называется автомат, у которого множество внутренних состояний и входных сигналов (а следовательно, и множество выходных сигналов) являются конечными множествами.*

Конечный автомат можно представить как математическую схему *(F-схему),* характеризующуюся шестью элементами: конечным множеством X входных сигналов; конечным множеством Y выходных сигналов; конечным множеством Z внутренних состояний; начальным состоянием z_0, $z_0 \in Z$; функцией переходов $\varphi(z,x)$; функцией выходов $\psi(z,x)$. Автомат, задаваемый *F-схемой* $F =< Z, X, Y, \varphi, \psi, z_0 >$ функционирует в дискретном автоматном времени, моментами которого являются такты, т. е. примыкающие друг к другу равные интервалы времени, каждому из которых соответствуют постоянные значения входного и выходного сигналов и внутренние состояния. Обозначим состояние, а также входной и выходной сигналы, соответствующие t такту при $t = 0, 1, 2,...$, через 0, $x(t)$, $y(t)$. При этом по условию $z(0)=z_0$,: $z \in Z$, $x \in X$, $y \in Y$. В момент t, будучи в состоянии $z(t)$, автомат способен воспринять на входном канале сигнал $x(t) \in X$ и выдать на выходном канале сигнал $y(t) = \psi(z(t), x(t)) \in Y$, переходя в состояние $z(t+1) = \varphi(z(t), x(t)) \in Z$. Если на вход конечного автомата, установленного в начальное состояние z_0, подавать в некоторой последовательности буквы входного алфавита $x(0), x(1), x(2),...$, т.е. входное слово, то на выходе автомата будут последовательно появляться буквы выходного алфавита $y(0)$, $y(1)$, $y(2),...$, образуя выходное слово. Это так называемый автомат первого рода, называемый также *автоматом Мили.* Автомат второго рода, у которого $y(t) = \psi(z(t)) \in Y$, т.е. функция выходов не зависит от входной переменной $x(t)$, называется *автоматом Мура.*

По числу состояний различают конечные автоматы с памятью и без памяти. Автоматы с памятью имеют более одного состояния, а автоматы без памяти (комбинационные или логические схемы) обладают лишь одним состоянием. При этом работа комбинационной схемы заключается в том, что она ставит в соответствие каждому входному сигналу $x(t)$ определенный выходной сигнал $y(t)$, т.е. реализует логическую функцию вида $y(t) = \varphi(x(t))$ $t = 0, 1, 2,...$.

Эта функция называется булевой, если алфавиты X и Y, которым принадлежат значения сигналов x и y, состоят из двух букв.

Простейший табличный способ задания конечного автомата основан на использовании таблиц переходов и выходов, строки которых соответствуют входным сигналам автомата, а столбцы − его состояниям. При этом обычно первый слева столбец соответствует начальному состоянию z_0. На пересечении x строки и z столбца таблицы переходов помещается соответствующее значение $\varphi(z, x)$ функции переходов, а в таблице выходов - соответствующее значение $\psi(z, x)$ функции выходов. Для автомата Мура обе таблицы можно совместить, получив так называемую отмеченную таблицу переходов, в которой над каждым состоянием z автомата, обозначающим столбец таблицы, стоит соответствующий этому состоянию выходной сигнал y.

Описание работы автомата Мили таблицами переходов $\varphi(z, x)$ и выходов $\psi(z, x)$ иллюстрируется табл. 5.1, а описание автомата Мура - таблицей переходов (табл. 5.2).

Таблица 5.1

Конечный автомат Мили

X	Z			
	z_0	$z1$	$z2...zm$-1	zm
Переходы				
$x1$	$\varphi(z_0, x_1)$	$\varphi(z_1, x_1)$	…	$\varphi(z_m, x_1)$
$x2$	$\varphi(z_0, x_2)$	$\varphi(z_1, x_2)$	…	$\varphi(z_m, x_2)$
xn	$\varphi(z_0, x_n)$	$\varphi(z_1, x_n)$	…	$\varphi(z_m, x_n)$
Выходы				
$x1$	$\psi(z_0, x_1)$	$\psi(z_1, x_1)$	…	$\psi(z_m, x_1)$
$x2$	$\psi(z_0, x_2)$	$\psi(z_1, x_2)$	…	$\psi(z_m, x_2)$
xn	$\psi(z_0, x_n)$	$\psi(z_1, x_n)$	…	$\varphi(z_m, x_n)$
	$Y1$	$y2$	$y3...yk$-1	yk
Y				

Таблица 5.2

Конечный автомат Мура

X	Z			
	z_0	$z1$	$z2...zm$-1	zm
$X1$	$\varphi(z_0, x_1)$	$\varphi(z_1, x_1)$	…	$\varphi(z_m, x_1)$
$X2$	$\varphi(z_0, x_2)$	$\varphi(z_1, x_2)$	…	$\varphi(z_m, x_2)$
xn	$\varphi(z_0, x_n)$	$\varphi(z_1, x_n)$	…	$\varphi(z_m, x_n)$
	$\psi(z_i)$			
	$Y1$	$y2$	$y3...yk$-1	yk
	Y			

5.12. Дискретно-вероятностные модели

В общем виде вероятностный автомат можно определить как дискретный потактный преобразователь информации с памятью, функционирование которого в каждом такте зависит только от состояния памяти в нем и может быть описано статистически. Рассмотрим множество G, элементами которого являются всевозможные пары (x, z), где x, z – элементы входного подмножества X и подмножества состояний Z соответственно. Если существуют две такие функции φ и ψ, и с их помощью осуществля-

ются отображения $G \to Z$ и $Z \to Y$, то говорят, что $P = <Z, X, Y, \varphi, \psi>$ определяет автомат детерминированного типа.

Более общая математическая схема строится следующим образом. Пусть Φ - множество всевозможных пар вида (zk, yj), где yk - элемент выходного подмножества Y. Потребуем, чтобы любой элемент множества G индуцировал на множестве Φ некоторый закон распределения следующего вида:

Элементы из Ф	• • •	$(x1, y1)$	• • •	$(x1, y2)$	• • •	(xK, yJ)
(zk, yj)	• • •	$b11$	• • •	$b12$	• • •	bKJ

При этом

$$\sum_{k=1}^{K} \sum_{j=1}^{J} b_{kj} = 1,$$

где b_{kj} — вероятности перехода автомата в состояние zk и появления на выходе сигнала yj, если он был в состоянии zk и на его вход в этот момент времени поступил сигнал xi . Число таких распределений, представленных в виде таблиц, равно числу элементов множества G. Обозначим множество этих таблиц через B. Тогда четверка элементов $P=< Z, X, Y, B >$ называется вероятностным автоматом (P-автоматом).

Пусть элементы множества G индуцируют некоторые законы распределения на подмножествах Y и Z, что можно представить соответственно в следующем виде:

Элементы из Y	• • •	$y1$	$y2$	• • •	yJ-1	yJ
(zs, xj)	• • •	$q1$	$q2$	• • •	$q\,J$-1	qJ

При этом

$$\sum_{j=1}^{J} q_j = 1,$$

где q_j − вероятности появления выходного сигнала yj .

Элементы из Z	• • •	$z1$	$z2$	• • •	zK-1	zK
(zs, xj)	• • •	$p1$	$p2$	• • •	pK-1	pK

При этом

$$\sum_{k=1}^{K} p_k = 1,$$

где p_k − вероятности перехода автомата в состояние zk . Для обеих таблиц выполняется условие, что P-автомат находился в состоянии zs и на его вход поступил входной сигнал xj.

Если для всех s и j имеет место соотношение $pkqj = bkj$, то такой P-автомат называется *вероятностным автоматом Мили*. Это требование означает выполнение условия независимости распределений для нового состояния P-автомата и его выходного сигнала.

Пусть теперь определение выходного сигнала *P*-автомата зависит лишь от того состояния, в котором находится автомат в данном такте работы. Другими словами, пусть каждый элемент выходного подмножества *Y* индуцирует распределение вероятностей выходов, имеющее следующий вид:

Элементы из *Y*	• • •	*y*1	*y*2	• • •	*yJ*-1	*yJ*
zs	• • •	*q*1	*q*2	• • •	*qJ*-1	*qJ*

Если для всех *s* и *j* имеет место соотношение *psqj* = *bsj*, то такой *P*-автомат называется *вероятностным автоматом Мура*. Частным случаем *P*-автомата являются автоматы, у которых либо переход в новое состояние, либо выходной сигнал определяются детерминированно. Если выходной сигнал определяется детерминированно, то такой автомат называется *Y*-детерминированным вероятностным автоматом. Аналогично *Z*-детерминированным вероятностным автоматом называется *P*-автомат, у которого выбор нового состояния является детерминированным.

Пример 1. Пусть задан *Y*-детерминированный *P*-автомат:

Z	*z*0	*z*1	*z*2	*z*3	*z*4
*z*0	0	0,5	0	0	0,5
*z*1	0	0	0	1	0
*z*2	0	0,7	0	0	0,3
*z*3	0	0	0,6	0	0,4
~~*z*4~~	~~0~~	~~1,0~~	~~0~~	~~0~~	~~0~~

Z	*z*0	*z*1	*z*2	*z*3	*z*4
Y	0	1	0	0	1

Требуется оценить суммарные финальные вероятности пребывания этого *P*-автомата в состояниях *z*1 и *z*4. При использовании аналитического подхода можно записать известные соотношения из теории марковских цепей и получить систему уравнений для определения финальных вероятностей. При этом начальное состояние *z0* можно не учитывать, так как начальное распределение не оказывает влияния на значения финальных вероятностей *Рф*. Тогда имеем:

$$P = \begin{matrix} 0 & 0 & 1 & 0 \\ 0,7 & 0 & 0 & 0,3 \\ 0 & 0,6 & 0 & 0,4 \\ 1,0 & 0 & 0 & 0 \end{matrix} \qquad P*Pф = Pф,$$

где *Рф* − матрица финальных вероятностей; *I* – единичная матрица. Тогда, решая систему уравнений, получим: $z1 = 0{,}3145$, $z2 = 0{,}1887$, $z3 = 0{,}3145$, $z4 = 0{,}1824$. Таким образом: $(z1 + z4) = 0{,}3145 + 0{,}1824 = 0{,}4969$.

5.1.3. Разработка моделей конечных автоматов в системе MATLAB

При моделировании конечного автомата сначала определяются входной X и выходной Y алфавиты, а также алфавит состояний автомата Z. Это могут быть числа натурального ряда, двоичные коды, символы, слова национальных алфавитов, специальные знаки и т.п. Для упрощения принимаются числа натурального ряда 1, 2, 3,... или их двоичные представления. Функции перехода задаются таблицами, формы которых были приведены в п. 5.1.1.

Пример 2. Составить *m*-функцию, моделирующую автомат Мура, при следующих значениях его параметров:
$X = \{1, 2, 3, 4, 5, 6\}$; $Z = \{1, 2, 3, 4\}$; $z0 = 0$; $Y = \{00, 01, 10, 11\}$ (табл. 5.3)

Таблица 5.3

X		*z*0	Z(1)	Z(2)	Z(3)	Z(4)
X(1)	1	1	1	2	3	4
X(2)	2	1	2	3	4	1
X(3)	3	1	3	4	3	4
X(4)	4	1	4	1	2	3
X(5)	5	1	1	1	1	1
		Y				
		00	00	01	10	11

```
function [z]=Fi(z0,x,Tabl)
% Функция перехода конечного автомата Мура
%[z]=Fi(z0,x,Tabl)
% z0 – начальное состояние
% x  - входной сигнал
% Tabl - таблица переходов размером (Nx * Nz)
z(1)=Tabl(x(1),z0);
l=length(x);
for i=2:l
  z(i)=Tabl(x(i),z(i-1));
end

function [y]=Psi(z)
% Функция выхода конечного автомата Мура
%[y]=Psi(z)
```

```
l=length(z);
y(1:2,1:l)=' ';
for i=1:l
  switch z(i)
  case 1,
    y(1,i)='0';y(2,i)='0';
  case 2,
    y(1,i)='0';y(2,i)='1';
  case 3,
    y(1,i)='1';y(2,i)='0';
  case 4,
    y(1,i)='1';y(2,i)='1';
  otherwise,
    y(1,i)='?';y(2,i)='?';
  end
end
```

Результаты моделирования при входном сигнале из 15 элементов:

```
x =          [ 1   2   3   4   5   1   2   3   4   5   4   3   2   1]
;
» z = fi(1, x, T);
z =           1   2   4   3   1   1   2   4   3   1   4   4   1   1
» y = psi(z);
y =           0   0   1   1   0   0   0   1   1   0   1   1   0   0
              0   1   1   0   0   0   1   1   0   0   1   1   0   0
```

Моделирование вероятностного автомата продемонстрируем на примере *Y*-детерминированного автомата из п. 5.1.2. Это дискретный односвязный скалярный марковский процесс с четырьмя возможными состояниями. Составим m-функцию, в которой по случайному числу от датчика равномерно распределённых чисел и вектору вероятностей находится случайный номер состояния процесса.

Пример 3. Проверка функции вероятностного перехода

```
function s=StepT(Pr)
% Функция выбора случайного состояния на один шаг
% s=StepT(x,Pr)
l=length(Pr);
x=rand(1);
h=0;
for i=1:l
  h=h+Pr(i);
  if x<h
    s=i;
    break
```

```
  end
end
```

	Z(1)	*Z*(2)	*Z*(3)	*Z*(4)
Вектор вероятностей:	0,7000	0	0	0,3000

Состояния *Z*(*t*) = 1 4 4 1 4 1 1 4 1 1 1 1 1 1 1 1 4 1 4 1

```
function M=Markow(z0,P,n)
% Генерация марковской последовательности
% M=Markow(z0,P,n)
% z0 – начальное состояние
% P  - вероятностная матрица переходов
% n - количество шагов
M(1)=stept(P(z0,:));
for i=2:n
   M(i)=stept(P(M(i-1),:));
end
```

Доля нахождения марковского процесса в каждом состоянии в численном эксперименте (100 временных шагов) очень близка к финальным вероятностям:

Состояния цепи	*Z*(1)	*Z*(2)	*Z*(3)	*Z*(4)
Финальные вероятности	0,3145	0,1887	0,3145	0,1824
Экспериментальные данные	0,32	0,19	0,32	0,17

5.2. Домашнее задание

1. Ознакомиться с формами представления моделей конечных автоматов, приведенных в п. 5.1.1 и 5.1.2.
2. Составить схему детерминированного конечного автомата с входным алфавитом из 3 символов, алфавитом состояний из 7 символов и выходным алфавитом из 2 символов, т.е. предложить функции перехода и выхода этой схемы.
3. Составить m-файл для моделирования конечного автомата, описанного в п.2 домашнего задания.
4. Составить схему вероятностного конечного автомата с входным алфавитом из 2 символов, алфавитом состояний из 10 символов и выходным алфавитом из 2 символов, т.е. предложить функции вероятностного перехода и выхода этой схемы.
5. Составить m-файл для моделирования конечного автомата, описанного в п. 4 домашнего задания.
6. Записать m-файл моделирования комбинационной схемы – двоичного дешифратора на 16 входов.

5.3. Лабораторное задание

1. Составить и провести моделирование комбинационной схемы – двоичного дешифратора на 16 входов.
2. По составленному в домашнем задании m-файлу смоделировать поведение детерминированного конечного автомата при подаче на его вход последовательности из 100 элементов входного алфавита. Начальное состояние выбрать первым из возможных. Вычислить автокорреляционную функцию и энергетический спектр выходного процесса и построить его гистограмму.
3. Повторить выполнение п. 1 лабораторного задания при других (остальных) начальных состояниях и той же входной последовательности. Провести сравнение автокорреляционных функций, энергетических спектров и гистограмм и сделать выводы о зависимости выходного сигнала от начального состояния.
4. По составленному в домашнем задании m-файлу смоделировать поведение вероятностного конечного автомата при подаче на его вход последовательности из 100 элементов входного алфавита. Начальное состояние выбрать первым из возможных. Вычислить автокорреляционную функцию и энергетический спектр выходного процесса и построить его гистограмму.
5. Повторить выполнение п. 3 лабораторного задания при других (остальных) начальных состояниях и той же входной последовательности. Провести сравнение автокорреляционных функций, энергетических спектров и гистограмм с финальными вероятностями и сделать выводы о зависимости выходного сигнала от начального состояния вероятностного конечного автомата.
6. Составить и провести моделирование комбинационной схемы – двоичного дешифратора.

5.4. Содержание отчёта

1. Цель работы.
2. Описание составленных в домашнем задании к лабораторной работе функций системы MATLAB: назначение, входные и выходные параметры, текст m-функции.
3. Результаты моделирования комбинационной схемы – двоичного дешифратора.
4. Описание моделируемого детерминированного конечного автомата: входной, выходной алфавиты, алфавит состояний, матрицы и графы перехода и выхода.
5. Графики входной и выходной последовательностей, график состояний автомата.
6. Результаты анализа выходной зависимости от начального состояния.

7. Описание моделируемого вероятностного конечного автомата: входной, выходной алфавиты, алфавит состояний, вероятностная матрица перехода и выхода.
8. Графики входной и выходной последовательностей, график состояний автомата. Гистограммы выходного процесса с финальными вероятностями.
9. Результаты анализа выходной зависимости от начального состояния.
10. Выводы по работе.

5.5. Контрольные вопросы

1. Какие формы представления конечных автоматов вы знаете?
2. Поясните различия в поведении детерминированного и вероятностного автоматов.
3. Как сказывается на работе автомата наличие или отсутствие памяти его состояний?
4. Приведите пример односвязной, 2-связной и 3-связной марковской последовательности.
5. Что такое *Y*-детерминированный вероятностный автомат? Чем он отличается от *Z*-детерминированного вероятностного автомата?
6. Какие динамические процессы можно представить марковскими моделями?
7. В чём отличие схемы моделирования коррелированной последовательности с помощью линейного фильтра от использования марковской модели?
8. Какую роль играют автокорреляционная функция и энергетический спектр в анализе процессов в конечных автоматах?
9. Может ли появиться тренд при моделировании конечных автоматов?
10. Какие функции системы MATLAB могут быть использованы для моделирования конечных автоматов?

СПИСОК ЛИТЕРАТУРЫ

1. Потёмкин, В.Г. Введение в MATLAB [Текст] / В.Г. Потемкин. − М.: Диалог-МИФИ, 2000. – 247с.

2. Баскаков, С.И. Радиотехнические цепи и сигналы [Текст] / С.И. Баскаков. − М.: Высш. шк., 1988. − 536 с.

3. Полляк, Ю.Г. Статистическое машинное моделирование средств связи [Текст] / Ю.Г. Поляк, В.А. Филимонов. – М.: Радио и связь, 1988. – 176 с.

4. Максимей, И.В. Имитационное моделирование на ЭВМ [Текст] / И.В. Максимей. – М.: Радио и связь, 1988.

5. Борисов, Ю.П. Цветнов В.В. Математическое моделирование радиотехнических систем и устройств [Текст] / Ю.П. Борисов. − М.: Радио и связь, 1985.

6. Дьяконов, В. MATLAB. Анализ, идентификация и моделирование систем [Текст]: специальный справочник / В. Дьяков, В. Круглов. СПб.: Питер, 2002. – 448 с.

7. Советов, Б.Я. Моделирование систем [Текст]: учебник для вузов / Б.Я. Совентов, С.А. Яковлев. – М.: Высшая школа, 1998. – 319 с.

8. Дьяконов, В.П. MATLAB 5.3.1 с пакетами расширений [Текст] / В.П. Дьяконов // под ред. проф. В.П. Дьяконова. − М.: Нолеж, 2001. – 880 с.

ОГЛАВЛЕНИЕ

В.Г. Лисиенко
С.П. Санников

МОДЕЛИРОВАНИЕ СИСТЕМ

Екатеринбург
2011